P.1877/90

6 1990

informatyka

Przetwarzanie zapytań w relacyjnych bazach danych

Programowanie EPROM

Ręczne rejestratory danych

WYDAWNICTWO MOT SIGMA

Cena 5800 zł

informatyka

Miesięcznik
ISSN 0542-9951. INDEX 36124

nr 6 1990
Czerwiec
Rok wydania XXV

W numerze:

	Strona
Jednoczesne przetwarzanie wielu zapytań w relacyjnych bazach danych – *Zbyszko Królikowski*	1
Metody programowania pamięci EPROM – *Piotr Orwaldi*	5
Ręczne rejestratory danych – *Jerzy Dawidowski*	8
Rozpoznawanie słów w mowie ciągłej – *Mieczysław Kłopotek*	12
CDS/ISIS narzędzie do zarządzania bazami danych (2) – *Tomasz Żakowicz*	15
Maszyny lispowe (2) *Jacek Malec*	19
Protokoły komunikacyjne (2). Model ISO – *Piotr Dembiński*	24
Ze świata	28
Ada dla mikrokomputerów MacIntosh	
Informatyka na wyższych uczelniach RFN	

KOLEGIUM REDAKCYJNE:

mgr Jarosław DEMINET,
dr inż. Wacław ISZKOWSKI,
mgr Teresa JABŁOŃSKA,
(sekretarz redakcji)
Władysław KLEPACZ
(redaktor naczelny)
dr inż. Wojciech MOKRZYCKI,
mgr inż. Jan RYŻKO,
mgr Hanna WŁODARSKA.

PRZEWODNICZĄCY
RADY PROGRAMOWEJ:

Prof. dr hab.
Juliusz Lech KULIKOWSKI

WYDAWCA: Wydawnictwo Czasopism i Książek Technicznych
SIGMA NOT
Spółka z o.o.
ul. Biała 4,
00-950 WARSZAWA
skrytka pocztowa 1004

Redakcja: 01-552 Warszawa
Pl. Inwalidów 10, p. 128, 136
tel. 39-14-34

Materiałów nie zamówionych redakcja nie zwraca

DRUK:
PRASOWE ZAKŁADY GRAFICZNE
RSW „PRASA-KSIĄŻKA-RUCH
ul. Dworcowa 13,
85-950 BYDGOSZCZ
zam. 1152/90
Obj. 4,0 ark. druk. Nakład 5300 egz.

Cena egzemplarza 5800 zł

W sprawach ogłoszeń prosimy zwracać się bezpośrednio do Redakcji

W najbliższych numerach:

- Adam Tucholski przedstawia zasady komunikacji między procesami użytkowymi trzech warstw architektury OSI/ISO dla sieci IBM PC Network.
- Dorota Szczawińska i Bogdan Trawiński charakteryzują oprogramowanie służące do wymiany danych między pakietami dBase III Plus oraz micro CDS/ISIS na mikrokomputerach typu IBM PC.
- Henryk Rybiński i Piotr Sobolewski opisują projekt systemu graficznego bazy danych GT-HOLMES.
- Nguyen Kim Sach omawia sprzętowe wzmacnianie obrazów cyfrowych w czasie rzeczywistym w celu poprawy ich jakości.
- Krzysztof Zieliński rozważa możliwość rozproszenia przetwarzania zadań graficznych oraz naświetla, jak zostały one wykorzystane w systemach graficznych budowanych przy użyciu transputerów.

WARUNKI PRENUMERATY

Prenumeratorzy zbiorowi jednostki gospodarki uspołecznionej, instytucje i organizacje społeczne zamawiają prenumeratę dokonując wpłat wyłącznie na blankiecie „wpłata zamówienie" (jest to „polecenie przelewu" rozszerzone dla potrzeb Wydawnictwa o część dotyczącą zamówienia). Blankiety te będą dostarczane dotychczasowym prenumeratorom przez Zakład Kolportażu. Nowi prenumeratorzy otrzymają je po zgłoszeniu zapotrzebowania (pisemne lub telefoniczne) w Zakładzie Kolportażu, w Radach Wojewódzkich NOT bądź w Redakcjach czasopism.

Prenumeratorzy indywidualni osoby fizyczne zamawiają prenumeratę dokonując wpłaty w UPT lub NBP na blankiecie NBP. Na odwrocie wszystkich odcinków blankietu należy wpisać tytuł czasopisma, okres prenumeraty, liczbę zamawianych egzemplarzy oraz wartość wpłaty. Wpłacać należy na konto: **Państwowy Bank Kredytowy III/O Warszawa nr 370015-7490-139-11.**

Prenumerata ulgowa przysługuje wyłącznie osobom fizycznym – członkom SNT, studentom i uczniom szkół zawodowych. Warunkiem prenumeraty ulgowej jest poświadczenie blankietu wpłaty (przed jej dokonaniem) na wszystkich odcinkach pieczęcią Koła SNT, wyższej uczelni lub szkoły. Sposób zamawiania prenumeraty ulgowej jest taki sam jak prenumeraty indywidualnej. W prenumeracie ulgowej można zamówić tylko po jednym egzemplarzu każdego czasopisma.

Uwaga! Miesięcznik „Aura" może być zamawiany w prenumeracie ulgowej również przez uczniów szkół ogólnokształcących.

Prenumerata ze zleceniem wysyłki za granicę zamawia się tak, jak prenumeratę indywidualną. Dodatkowo należy podać na blankiecie wpłaty nazwisko i dokładny adres odbiorcy.

Cena prenumeraty ze zleceniem wysyłki za granicę jest dwukrotnie wyższa.

Wpłaty na prenumeratę przyjmowane są w terminach:
do 10 listopada na każdy kwartał, I i II półrocze oraz cały rok następny,
do 28 lutego na II, III i IV kwartał oraz II półrocze,
do 31 maja na III i IV kwartał oraz II półrocze,
do 31 sierpnia na IV kwartał.

Zmiany w prenumeracie można zgłaszać pisemnie tylko w wyżej wymienionych terminach.

Informacji o prenumeracie udziela Zakład Kolportażu Wydawnictwa NOT SIGMA (ul. Bartycka 20, 00-716 Warszawa), skr. poczt. 1004, 00-950 Warszawa, tel. 40-00-21 wew. 248, 249, 293, 297, 299 lub 40-30-86 i 40-35-89.

Egzemplarze archiwalne czasopism – można nabyć za gotówkę w Klubie Prasy Technicznej, Warszawa ul. Mazowiecka 12 (tel. 26-80-16), lub zamówić pisemnie. Zamówienia na egzemplarze archiwalne czasopism przyjmuje: Zakład Kolportażu, Dział Handlowy, 00-950 Warszawa, skr. poczt. 1004 (tel. 40-37-31), na rachunek dla instytucji lub za zaliczeniem pocztowym dla osób fizycznych.

Cena egzemplarza: normalna 5800 zł, ulgowa 1160 zł

Uwaga! Podane ceny mają charakter wstępny i mogą ulec zmianie, w związku z powyższym Wydawnictwo zastrzega sobie prawo żądania dopłat do już opłaconej prenumeraty.

P.1877/90

ZBYSZKO KRÓLIKOWSKI
Instytut Informatyki, Automatyki i Robotyki
Politechnika Poznańska

Jednoczesne przetwarzanie wielu zapytań w relacyjnych bazach danych

Historia relacyjnych baz danych ma już 19 lat, licząc od opublikowania przez E. F. Codd'a fundamentalnej pracy [6]. W ostatnich kilku latach prowadzi się intensywne prace nad systemami rozproszonych relacyjnych baz danych. Najogólniej przez *system rozproszonej bazy danych* (SRBD) rozumie się system złożony z pewnej liczby niezależnych, rozproszonych geograficznie stanowisk komputerowych, połączonych liniami transmisyjnymi, które umożliwiają wymianę informacji między nimi. Zainteresowanie systemami RBD wynika z tego, że dopiero realizacja takich systemów stanowi podstawę do budowy systemów użytkowych dla bardzo wielu zastosowań, które z natury rzeczy mają zdecentralizowany charakter (np. systemy rezerwacji lotniczej, hotelowej, systemy bankowe, pocztowe, systemy zarządzania przedsiębiorstwami wielozakładowymi itp.). Uzyskanie potencjalnych korzyści wynikających z realizacji systemów RBD wymaga jednak rozwiązania wielu jakościowo nowych, trudnych problemów. Jednym z nich jest optymalizacja przetwarzania zapytań. Problem ten był już szczegółowo analizowany na łamach Informatyki w cyklu artykułów [9].

Wspólną cechą zdecydowanej większości proponowanych rozwiązań problemu optymalizacji wykonywania zapytań w SRBD jest rozpatrywanie środowiska jednozapytaniowego (patrz [9]). To znaczy, problem optymalizacji jest formułowany tak, jakby tylko jedno zapytanie przebywało w danej chwili w systemie bazy danych. Jednak optymalizacja każdego zapytania z osobna nie musi prowadzić do optymalizacji działania całego systemu. Wynika to z faktu, że w rzeczywistości system bazy danych, a w szczególności system rozproszonej bazy danych stanowi środowisko wielo-, a nie jednozapytaniowe, w którym w każdej chwili wiele zapytań konkuruje ze sobą w dostępie do tych samych zasobów systemu. Co więcej, zapytania zgłoszone do systemu bazy danych mogą zawierać wspólne elementy. Ilustruje to przykład przedstawiony poniżej.

Przykład

Niech będzie dana relacyjna baza danych opisująca zasoby biblioteki oraz dokumentująca jej działalność. Baza danych składa się z następujących relacji:

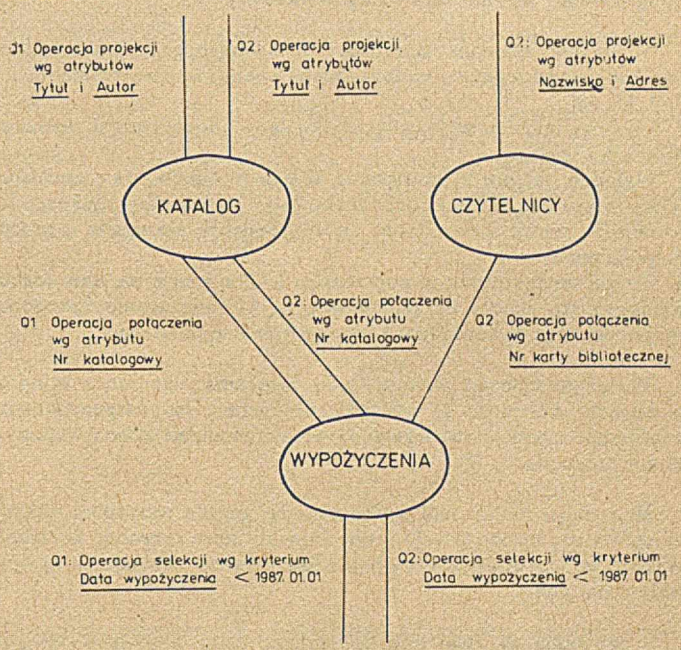
*R*1: **KATALOG** (*Tytuł, Autor, Wydawnictwo, Nr katalogowy*),
*R*2: **CZYTELNICY** (*Nazwisko, Adres, Nr karty bibliotecznej*),
*R*3: **WYPOŻYCZENIA** (*Nr katalogowy, Nr karty bibliotecznej, Data wypożyczenia*).

Do systemu zarządzania bazą danych (SZBD) zgłoszono dwa następujące zapytania (tutaj sformułowane w języku naturalnym):

*Q*1: *Podaj tytuły i autorów książek wypożyczonych przed 1987.01.01.*
*Q*2: *Podaj nazwiska i adresy czytelników oraz tytuły i autorów książek wypożyczonych przez te osoby przed 1987.01.01.*

Na rysunku 1. przedstawiono schematycznie wykonanie zapytań *Q*1 i *Q*2 w rozbiciu na elementarne operacje relacyjne projekcji, selekcji i połączenia. Jak wynika z tego schematu, zapytania *Q*1 i *Q*2 zawierają wiele wspólnych operacji cząstkowych.

Klasyczne podejście do problemu przetwarzania zapytań w systemach RBD polega na optymalizacji każdego zapytania osobno i wykonywaniu tych zapytań sekwencyjnie. W takiej sytuacji wspólne operacje cząstkowe zapytań *Q*1 i *Q*2 z przykładu będą wykonywane w systemie dwukrotnie: w trakcie realizacji zapytania *Q*1 i ponownie przy realizacji zapytania *Q*2. W nowym kierunku badawczym w problematyce optymalizacji przetwarzania zapytań w systemach relacyjnych baz danych uwzględnia się wielozadaniowy charakter systemu i zakłada, że w systemie jest wykonywanych jednocześnie wiele zapytań (ang. *multi-query*). Nowe podejście do optymalizacji przetwarzania zapytań polega więc na ich grupowaniu i minimalizowaniu kosztów wykonywania całych grup pytań. Zgodnie z tym podejściem w procesie optymalizacji, zapytania *Q*1 i *Q*2 z przykładu będą rozpatrywane łącznie. Do zredukowania kosztów wykonywania zbioru zapytań mogą być wykorzystane następujące wspólne elementy: operacje relacyjne projekcji, selekcji i połączenia, a w systemach rozproszonych baz danych – również operacja półpołączenia i transmisje danych. Dzięki grupowaniu zapytań wymienione powyżej wspólne dla wielu pytań operacje mogą być wykonywane tylko raz. Może to spowodować znaczne zmniejszenie kosztów lokalnego przetwarzania m.in. dzięki zmniejszeniu liczby fizycznych dostępów do plików poszczególnych relacji. W systemach RBD dodatkowo można uzyskać zmniejszenie kosztów transmisji, gdyż takie wyniki wspólnych operacji cząstkowych mogą być tylko jeden raz transmitowane.



Rys. 1. Przykładowe zapytania *Q*1 i *Q*2 zawierające wspólne składowe operacje relacyjne

W literaturze [3, 5, 8, 12] zaproponowano dotychczas zaledwie kilka częściowych rozwiązań problemu jednoczesnego przetwarzania wielu zapytań w systemach RBD. Rozwiązania te dotyczą przede wszystkim scentralizowanych baz danych; zakłada się w nich, że zapytania są przetwarzane w trybie wsadowym. Dlatego są one nieodpowiednie dla środowiska systemów rozproszonych baz danych. W tej pracy przedstawiono nową metodę jednoczesnego przetwarzania wielu zapytań, przeznaczoną dla środowiska systemów rozproszonych baz danych, oraz omówiono problemy związane z implementacją tej metody.

PODSTAWOWE POJĘCIA I ZAŁOŻENIA ORAZ CHARAKTERYSTYKA PROBLEMU

Problem opracowania efektywnego algorytmu wykonywania zapytań jest jednym z najtrudniejszych i najbardziej złożonych problemów

stojących przed projektantem systemu zarządzania RBD. Problem ten należy do klasy problemów silnie NP-zupełnych. Dotychczas opracowano wiele heurystycznych algorytmów przetwarzania zapytań w SRBD (m.in. [1, 2, 4, 9, 14, 15, 16, 17]), dotyczących środowiska jednozapytaniowego. W tych algorytmach szczególną uwagę poświęca się sposobom wykonywania operacji połączenia. Wynika to z faktu, że koszty wykonywania tej operacji w decydującym stopniu determinują koszt wykonywania zapytania w systemie bazy danych, zarówno scentralizowanym, jak i rozproszonym. W systemie RBD operacja połączenia wymaga zarówno transmisji dużych woluminów danych (jeśli łączone relacje znajdują się na różnych stanowiskach komputerowych), jak i czasochłonnego przetwarzania. W większości algorytmów przetwarzania zapytań w SRBD operację połączenia realizuje się wykonując najpierw odpowiednią sekwencję operacji półpołączenia (ang. *semijoin*). Operacja półpołączenia [2, 9] relacji Ri oraz relacji Rk, oznaczana tutaj $Ri \rightarrow Rk$, jest kompozycją operacji relacyjnych projekcji i połączenia. Po wykonaniu projekcji relacji Ri według atrybutu połączeniowego, wszystkie unikalne wartości atrybutu połączeniowego (stanowiące wynik) są łączone z odpowiednimi krotkami relacji Rk. Rozmiar relacji Rk zostaje w ten sposób zredukowany do tych tylko krotek, które biorą bezpośredni udział w relacji wynikowej połączenia obu relacji Ri i Rk.

Konsekwencją wspomnianego szczególnego znaczenia operacji półpołączenia jest powszechnie przyjmowany model zapytania. Zakłada się, że każde zapytanie Q zawiera dwa następujące składniki:

- kwalifikację zapytania $\mathbf{K}(Q)$ (ang. *qualification*), będącą koniunkcją operacji połączenia w postaci $Ri.Ai_1 = Rj.Aj_1$, to znaczy $\mathbf{K}(Q) = ((Rk.Ak_1 = Rl.Al_1)...(Ru.Au_1 = Rv.Av_1))$, przy czym pary atrybutów (Ak_1, Al_1),..., $(Au_1, Av_1$ są zdefiniowane na wspólnych dziedzinach;

- specyfikację relacji wynikowej $\mathbf{T}(Q)$ (ang. *target list*), zawierającą listę atrybutów, których wartości spełniające warunki określone w zapytaniu powinny się znaleźć w relacji wynikowej.

W dotychczasowych algorytmach przetwarzania zapytań w SRBD przyjmuje się następujący ogólny schemat wykonywania zapytania [1, 2, 7, 9, 13, 16].
1. Wykonaj operacje wymagające tylko danych lokalnych, tj. operacje projekcji i selekcji.
2. Wykonaj wstępne przetwarzanie, w celu zredukowania rozmiarów relacji przetwarzanych w następnym kroku (w tej fazie do zredukowania rozmiarów relacji wykorzystuje się odpowiednie sekwencje operacji półpołączenia).
3. Prześlij zredukowane w poprzednim kroku relacje na stanowisko wynikowe (to znaczy na stanowisko, na którym zostało zgłoszone zapytanie) oraz wykonaj tam końcową operację połączenia.

Powyższy schemat wykonywania zapytania, zgodnie z którym w pierwszej kolejności są wykonywane operacje jednoargumentowe, a następnie operacje dwuargumentowe, minimalizuje koszty wykonywania zapytania [1, 2].

W ramach badań nad nową metodą, przede wszystkim należy określić kiedy optymalizacja grupy zapytań może przynieść korzyści. Wydaje się, że jednoczesna optymalizacja wielu zapytań może być efektywna, jeśli:

– zapytania są zgłaszane do systemu bazy danych w sposób naturalny jako grupa, to znaczy w trybie wsadowym lub w trybie interakcyjnym, gdy użytkownik zgłasza do systemu zbiór zapytań jako jedną całość;

– zapytania są zgłaszane do systemu w trybie interakcyjnym, w krótkich odstępach czasu, tj. takich, że grupowanie zapytań nie ma zauważalnego dla użytkownika wpływu na czas wykonywania poszczególnych zapytań;

– pojedyncze zapytanie skierowane do systemu jest rozbijane na poziomie konceptualnym na kilka podzapytań, które mogą być przetwarzane jednocześnie.

Zakres możliwości stosowania metody jednoczesnej optymalizacji wielu zapytań jest więc dość szeroki. Jednak uzyskanie korzyści z zastosowania tej metody w praktyce wymaga rozwiązania kilku nowych problemów.

OPTYMALIZACJA PRZETWARZANIA WIELU ZAPYTAŃ W SRBD – PROPOZYCJE NOWYCH ROZWIĄZAŃ

Jednym z podstawowych problemów związanych z optymalizacją przetwarzania wielu zapytań w SRBD jest określenie liczby zapytań, które powinny być grupowane, aby wygenerowany wspólny plan przetwarzania był efektywny ze względu na przyjęte kryterium oceny działania systemu.

Grupowanie pytań

W systemach baz danych, w których przetwarzanie zapytań jest wykonywane w trybie wsadowym, korzyści z grupowania zapytań powiększają się wraz ze zwiększaniem rozmiaru grupy wspólnie optymalizowanych zapytań. W interakcyjnym SRBD rozmiar grupy jest jednak ograniczony przez maksymalny czas odpowiedzi, akceptowany przez pierwsze zapytanie z grupy zapytań zgłoszonych do systemu. Czas odpowiedzi jest definiowany jako czas upływający od chwili zgłoszenia zapytania przez użytkownika do uzyskania przez niego kompletnej odpowiedzi. Czas odpowiedzi dla pierwszego zapytania z grupy w interakcyjnym SRBD nie może ulec wydłużeniu w znaczący sposób. Innymi słowy, ewentualna zmiana tego czasu nie może być zauważalna dla użytkownika. Interpretacja określenia „znaczący sposób" zależy od typu i przeznaczenia systemu bazy danych.

Liczba zapytań, które mogą być grupowane, zależy również od rodzaju i stopnia złożoności zapytań zgłoszonych do systemu. Istotnym ograniczeniem mogą być również rozmiary pamięci buforowych poszczególnych stanowisk komputerowych SRBD wykorzystywanych w trakcie przetwarzania.

Przy określaniu liczby zapytań, które powinny być grupowane w celu jednoczesnego przetwarzania, należy więc brać pod uwagę wiele różnorodnych czynników. Możliwość znalezienia odpowiedniego rozwiązania tego problemu w drodze rozważań teoretycznych wydaje się co najmniej wątpliwa. Jedynym nasuwającym się rozwiązaniem [11] jest eksperymentalne określenie w konkretnym systemie maksymalnej „opłacalnej" różnicy czasu między zgłoszeniem do systemu pierwszego i ostatniego zapytania, które powinny się znaleźć w tej samej grupie. Te przedziały czasu powinny być określone dla wielu kombinacji różnych typów zapytań, przy uwzględnieniu wpływu obciążenia systemu. Przeprowadzenie odpowiednich badań eksperymentalnych nie jest zadaniem prostym, jednak możliwym do wykonania. Złożoność odpowiedniego eksperymentu zależy od sposobu klasyfikacji zapytań.

W proponowanym tutaj rozwiązaniu procedury optymalizacyjne systemu zarządzania RBD muszą dysponować dodatkowymi informacjami statystycznymi, zbędnymi w klasycznym podejściu do optymalizacji przetwarzania zapytań w SRBD. Co więcej, te dodatkowe informacje muszą być w systemie zabierane i co najmniej okresowo aktualizowane. Jeszcze kilka lat temu, takie wymagania należałoby uznać za bardzo poważną wadę proponowanego rozwiązania. Obecnie jednak, przy coraz korzystniejszych parametrach sprzętu, to dodatkowe obciążenie systemu jest pomijalne [10, 15, 17], a wykorzystanie informacji statystycznych do optymalizacji przetwarzania zapytań umożliwia dalszą poprawę efektywności działania SRBD.

Prostym rozwiązaniem problemu dysponowania pewnymi informacjami dotyczącymi charakterystyki poszczególnych zapytań może być przypisanie do każdego zapytania (przez procedurę SZRBD zwaną analizatorem zapytań) rekordu informacyjnego o następującej strukturze [11]:

Identyfikator użytkownika	*Identyfikator zapytania*	*Czas zgłoszenia zapytania*	*Typ zapytania*	*Kwalifikacja zapytania*	*Specyfikacja relacji wynikowej*

W metodzie jednoczesnego przetwarzania wielu zapytań procedury optymalizacyjne SZRBD muszą dysponować informacją umożliwiającą określenie, do którego zapytania z grupy należy dana odpowiedź. W tym celu z przedstawionego powyżej rekordu informacyjnego zapytania mogą być wykorzystane atrybuty *Identyfikator użytkownika* i *Identyfikator zapytania*. Atrybuty *Czas zgłoszenia zapytania* oraz *Typ zapytania* są wykorzystywane przy podejmowaniu decyzji, czy dane zapytanie powinno być włączone do grupy jednocześnie przetwarza-

nych zapytań. Oczywiście przy podejmowaniu tej decyzji procedury optymalizacyjne SZRBD biorą pod uwagę czas zgłoszenia pierwszego zapytania oraz „opłacalny" dla grupowania przedział czasu, otrzymany eksperymentalnie. Odpowiedni przedział czasu jest dobierany w zależności od typu zapytania i aktualnego obciążenia systemu. Atrybuty *Kwalifikacja zapytania* i *Specyfikacja relacji wynikowej* są wykorzystywane przez procedury optymalizacyjne SZRBD przy generowaniu planu jednoczesnego wykonywania grupy zapytań.

Migracja zapytań

W dotychczasowych metodach przetwarzania zapytań w SRBD przyjmuje się opisany wcześniej trzyetapowy schemat wykonywania zapytania. W proponowanym nowym rozwiązaniu dotyczącym optymalizacji przetwarzania wielu zapytań, zapytania są realizowane w czterech etapach – przed etapem wstępnego lokalnego przetwarzania proponuje się dodanie tak zwanego *etapu migracji zapytań* [11]. Celem tego etapu jest wybranie dla każdego zapytania stanowiska, do którego powinno być ono wysłane ze stanowiska, na którym zostało zgłoszone, tak aby możliwość dzielenia operacji relacyjnych przez zapytania z grupy jednocześnie przetwarzanych zapytań była maksymalna. Można to osiągnąć w następujący sposób.

W fazie migracji, każde stanowisko *i* systemu rozproszonej bazy danych jest traktowane jako stanowisko „wirtualne", na którym są przechowywane relacje fizycznie zlokalizowane na stanowisku *i* oraz relacje wymagane przez zapytanie, które w fazie migracji są traktowane jako wirtualnie zlokalizowane na tym stanowisku.

Dla każdego zapytania zgłoszonego do systemu należy sprawdzić, które stanowisko wirtualne zawiera maksymalną liczbę relacji wymaganych w kwalifikacji zapytania. Jeśli nie jest to stanowisko, z którego dane zapytanie zostało zgłoszone do systemu, to zapytanie należy przesłać do przetworzenia na wybrane stanowisko wirtualne. Ponieważ wszystkie stanowiska komputerowe, na których są zgłaszane zapytania, mają takie wirtualne spojrzenie na rozproszoną bazę danych, więc przez fazę migracji zwiększa się prawdopodobieństwo, że lokalnie na wybranym wirtualnym stanowisku przetwarzania zapytania będą dzielić wspólne, składowe operacje relacyjne.

Z powodów omówionych wcześniej, głównym punktem zainteresowania optymalizacji przetwarzania zapytań jest operacja połączenia. Tak też jest w przypadku metody tutaj proponowanej. Warto zwrócić uwagę, że wykonanie fazy migracji zapytań zwiększa również prawdopodobieństwo dzielenia przez wiele zapytań operacji selekcji i projekcji, to znaczy operacji wykonywanych w następnej fazie przetwarzania zapytań.

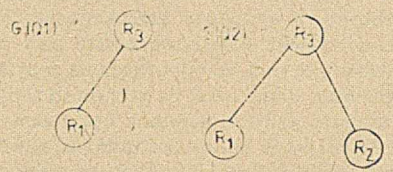
Faza wstępnego przetwarzania (tj. punkt 2 w prezentowanym wcześniej ogólnym schemacie wykonywania zapytań w SRBD) jest od kilku lat przedmiotem intensywnych badań. W zdecydowanej większości proponowanych dotychczas rozwiązań zakłada się wstępną redukcję rozmiarów łączonych relacji przed ich przesłaniem na stanowisko wynikowe i wykonaniem tam końcowej operacji połączenia. W celu zredukowania rozmiarów relacji wykorzystuje się odpowiednią sekwencję operacji półpołączenia [2]. Jednak operacja półpołączenia nie jest odpowiednim narzędziem redukcji rozmiarów łączonych relacji w systemach RBD implementowanych w lokalnych sieciach komputerowych, ani w rozległych sieciach komputerowych, dla małych rozmiarów łączonych relacji [10]. Zastosowanie tam operacji półpołączenia może spowodować nieprzewidziane zwiększenie kosztów wykonywania zapytania w porównaniu z metodami wykorzystującymi tylko operację połączenia. Powyższe wyniki zostały ostatnio potwierdzone przez pomiary przeprowadzone w jednym z najbardziej zaawansowanych, eksperymentalnych systemów rozproszonych baz danych – MERMAID (University of Chicago at Illinois) [17].

We współczesnych SRBD najbardziej odpowiednia jest *mieszana strategia wykonywania zapytania*, w której stosuje się operację połączenia lub półpołączenia, w zależności od tego, która z nich jest bardziej efektywna ze względu na przyjęte kryterium oceny działania systemu. Taką strategię zaprezentowano w [9], podobne rozwiązania wdraża się obecnie w systemach MERMAID [17] i GENEZIS [13].

Identyfikowanie wspólnych operacji

Niezależnie od tego, czy w tej fazie wykonywania zapytania zostanie przyjęta strategia operacji połączenia czy półpołączenia, w środowisku wielozapytaniowym procedury optymalizacyjne SZRBD muszą dysponować narzędziem, które pozwoli zidentyfikować operacje połączenia lub półpołączenia wspólne dla całej grupy jednocześnie wykonywanych zapytań. W tym celu można zastosować technikę grafów.

Dla dowolnego zapytania Q, na podstawie jego kwalifikacji $\mathbf{K}(Q)$ można skonstruować tak zwany graf zapytania $\mathbf{G}(Q)$. Zbiór wierzchołków grafu $\mathbf{G}(Q)$ odpowiada zbiorowi relacji ujętych w kwalifikacji zapytania, a krawędzie grafu odpowiadają operacjom połączenia na zbiorze wspólnych atrybutów połączeniowych tych relacji. Przykładowo, na rys. 2. przedstawiono grafy zapytań $Q1$ i $Q2$ z przykładu prezentowanego we wstępie tej pracy.



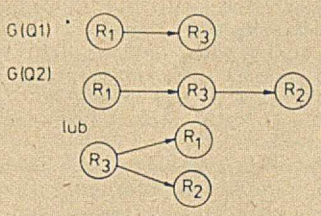
Rys. 2. Grafy przykładowych zapytań $Q1$ i $Q2$

Jak wspomniano wcześniej, w celu zredukowania rozmiarów relacji, których transmisja w systemie jest niezbędna do wykonania końcowej operacji połączenia, wykorzystuje się odpowiednią sekwencję operacji półpołączenia. Dla zapytań $Q1$ i $Q2$ z naszego przykładu mogą to być następujące sekwencje operacji półpołączenia:

$Q1$: $R1 \rightarrow R3$
$Q2$: $R1 \rightarrow R3$, $R3 \rightarrow R2$ lub $(R3 \rightarrow R1)$ $(R3 \rightarrow R2)$

Dla zapytania $Q2$, operacje półpołączenia ujęte w nawiasy mogą być wykonywane równolegle.

W podobny sposób sekwencja operacji półpołączenia może być reprezentowana w postaci grafu. Jednak w tym przypadku, ponieważ operacja półpołączenia jest operacją asymetryczną (tzn. $R1 \rightarrow R2 \neq R2 \rightarrow R1$), musi to być acykliczny graf skierowany. Jego wierzchołki reprezentują relacje, a skierowane krawędzie – odpowiednie operacje półpołączenia (rys. 3).



Rys. 3. Grafy skierowane sekwencji operacji półpołączenia przykładowych zapytań $Q1$ i $Q2$

Wspólne operacje dla całej grupy jednocześnie przetwarzanych zapytań mogą być identyfikowane w prosty sposób przez złożenie grafów wszystkich zapytań. W tym celu graf każdego zapytania powinien być reprezentowany w postaci relacji binarnej, wskazującej powiązania *rodzice–dzieci*. Oto reprezentacja grafów sekwencji operacji półpołączenia zapytań $Q1$ i $Q2$ z rysunku 3, w postaci odpowiednich relacji binarnych.

$Q1$:	*Rodzice*	*Dzieci*
	R1	*R3*

$Q2$:	*Rodzice*	*Dzieci*
	R1	*R3*
	R3	*R2*

lub

$Q2$:	*Rodzice*	*Dzieci*
	R3	*R1*
	R3	*R2*

Dysponując taką reprezentacją grafów zapytań, procedury optymalizacyjne mogą w prosty sposób stwierdzić, że np. dla grupy zapytań $Q1$ i $Q2$ wspólna jest operacja $R1 \rightarrow R3$. Operacja ta może być wykonana tylko raz, a jej wynik może być użyty do realizacji zarówno zapytania $Q1$, jak i $Q2$.

* * *

Przedstawiona propozycja zmodyfikowania ogólnego schematu wykonywania zapytań w SRBD (przez wprowadzenie fazy migracji zapytań) oraz propozycja wykorzystania grafów do łatwego identyfikowania wspólnych operacji składowych, pozwalają w sposób prosty i jednocześnie efektywny rozwiązać problem generowania planów wykonywania grupy zapytań. W proponowanym tutaj rozwiązaniu przyjmuje się, że dla każdego indywidualnego zapytania został już wygenerowany plan jego wykonywania metodą tradycyjną, tj. przy założeniu, że zapytania są wykonywane jedno po drugim. Tak więc, rozwiązanie to ma w pewnym sensie charakter uniwersalny, ponieważ umożliwia przetwarzanie zapytań albo metodą tradycyjną, albo metodą jednoczesnego przetwarzania wielu zapytań. Ta dwuwariantowość rozwiązania jest bardzo cenna ze względu na to, że metoda jednoczesnego przetwarzania grupy zapytań może, lecz nie musi spowodować zmniejszenie kosztów przetwarzania zapytań w porównaniu z dotychczasowymi sekwencyjnymi metodami przetwarzania. Faza migracji zapytań wprowadza bowiem dodatkowe koszty przy przetwarzaniu grupy zapytań, wynikające stąd, że odpowiedź na zapytanie musi być przesłana z powrotem na stanowisko, na którym to zapytanie zostało zgłoszone. Faza migracji zapytań powinna być zatem wykonywana tylko wówczas, gdy dodatkowe koszty związane z jej wykonywaniem są zrekompensowane przez redukcję kosztów wynikającą z grupowania zapytań. Ponadto, generowanie planu przetwarzania grupy zapytań jest bardziej czasochłonne w porównaniu z metodami sekwencyjnymi, ze względu na konieczność identyfikowania wspólnych operacji. Oznacza to, że przed podjęciem ostatecznej decyzji dotyczącej sposobu wykonywania grupy zapytań, konieczne jest oszacowanie ogólnych kosztów jednoczesnego i sekwencyjnego przetwarzania. Procedury optymalizacyjne SZRBD będą więc musiały dysponować odpowiednim modelem kosztów, na podstawie którego będzie możliwe takie oszacowanie.

Według szacunkowych ocen przedstawionych w [5, 8, 12], wdrożenie do praktyki metody jednoczesnego przetwarzania wielu zapytań, może przynieść oszczędności rzędu 60%, w porównaniu z kosztami wykonywania zapytań tradycyjną metodą sekwencyjną.

LITERATURA

[1] Apers P.M.G., Hevner A.R., Yao S.B.: Optimization algorithms for distributed queries. IEEE Trans. on Software Engineering, Vol. SE-9, No.1, pp. 57–68, 1983

[2] Bernstein P.A., Chiu D.W.: Using semijoins to solve relational queries. Journal of ACM, Vol. 28, No.1, pp. 25–40, 1981

[3] Cellary W., Meyer D.: A multi-query approach to distributed processing in a relational distributed data base management system. In: Distributed Data Bases (Eds. C. Delobel, W. Litwin), North-Holland, pp. 99–119, 1980

[4] Cellary W., Królikowski Z., Morzy T.: Other comments on „Optimization Algorithms for Distributed Queries". IEEE Trans. on Software Engineering, Vol. 14, No. 4, pp. 439–442, 1988

[5] Chesnais A., Gelenbe E., Mitrani J.: On the modelling of parallel access to shared data. Communication of the ACM, Vol. 26, No. 3, pp. 196–202, 1983

[6] Codd E. F.: A relational model of data for large shared data banks. Communications of the ACM, No. 13, pp. 377–387, 1970

[7] Hevner A.R., Wu C., Yao S.B.: Query optimization on local area network. ACM Trans. on Office Information Systems, Vol. 3, No. 1, pp. 35–62, 1985

[8] Jarke M.: Common subexpresion isolation in multiple query optimization. In: Query Processing in Database Systems (Eds. W. Kim, D.S. Reiner, D.S. Batory), Springer-Verlag, pp. 191–206, 1985

[9] Królikowski Z.: Algorytmy generowania planów wykonywania transakcji w systemach rozproszonych baz danych. Cz. 1, 2, 3. Informatyka, Nr 6, 7–8, 9–10, 1986

[10] Królikowski Z., Li V.O.K.: Effectiveness of distributed query strategies in wide and local area networks. Technical Report – Department of Electrical Engineering, University of Southern California (CSI-11/18/87), Los Angeles, 1987

[11] Królikowski Z., Li V.O.K., Huang C.L.: Multiquery processing in distributed database systems. Proc. of 10th Intern. Symposium – Computer at the University, Dubrovnik, Jugosławia, 1988

[12] Kim W.: Global optimization of relational queries – a first step. In: Query Processing in Database Systems (Eds. W. Kim, D.S. Reiner, D.S. Batory), Springer-Verlag, pp. 206–216, 1985

[13] Page T.W., Popek G.J.: Distributed data management in local area networks. Proc. of the 4th ACM-SIGACT-SIGMOD Symposium on Principles of Database Systems, Portland, Oregon, pp. 135–142, 1985

[14] Perrizo W.K.: Distributed query processing algorithm yielding a least upper bound response time semijoin strategies. Proc. of the ist Intern. Conf. on Supercomputer Systems, St. Petersburg, Florida, pp. 273–279, 1985

[15] Su S.Y.W., et al: A distributed query processing strategy using decomposition, pipelining and intermediate result sharing techniques. Proc. of the Intenr. Conf. on Data Engineering, Los Angeles, pp. 94–102, 1986

[16] Yu C.T., Chang C.C.: Distributed query processing. ACM Computer Surveys, Vol. 16, No. 4, pp. 400–433, 1984

[17] Yu C.T., et al: Algoritms to process distributed queries in fast local networks. IEEE trans. on Computers, Vol. C-36, No. 10, pp. 1155–1164, 1987.

Micomp

ZAKŁAD SYSTEMÓW MIKROKOMPUTEROWYCH
MICOMP

40–045 Katowice, ul. Astrów 7
telefon i telefaks: 518-628
teleks: 315687

Systemy teletransmisji danych
ICL 1900, 2900, ME 29, 39,
ODRA 1305, EMCX 1305:

skaner **MPXSCAN-8007**,
procesor sieci WAN
MICOMP-8075
(emulacja ICL 7503),
program teletransmisji danych
MICROS FXBM
(PC, rozproszone bazy danych,
wersja sieciowa – NOVELL),
adaptery, modemy, testery,
emulatory terminali

Integracja systemów ICL i ODRA
z mikrokomputerami standardu PC
i systemami NOVELL, UNIX SCO

Instalacje w największych systemach
na terenie całego kraju!

I.O 1124 89

PIOTR ORWALDI
Wrocław

Metody programowania pamięci EPROM

Rozwój elektroniki i informatyki spowodował zwiększenie zapotrzebowania na układy pamięci stałych o różnych pojemnościach. Dzięki postępowi technologii elektronowej przodujące firmy elektroniczne oferują duży wybór reprogramowalnych pamięci stałych PROM. Artykuł ma na celu przybliżenie tematyki programowania reprogramowalnych pamięci stałych EPROM.

Pamięć stała cechuje się tym, że raz zapisana informacja jest trwale przechowywana przez długi czas i może być wielokrotnie odczytywana. Tego rodzaju pamięci są stosowane np. do przechowywania systemów operacyjnych, generatorów znaków itp. Pamięci stałe można ogólnie podzielić na dwie grupy:

- pamięci programowane w sposób trwały bez możliwości wymazania raz zapisanej informacji (PROM),
- pamięci programowane w sposób prawie trwały z możliwością wymazania i ponownego zapisu informacji (pamięci reprogramowalne – EPROM, EEPROM).

Dalszy podział wynika ze sposobu wymazywania informacji w pamięciach drugiej grupy. Pamięci reprogramowalne można podzielić na dwa rodzaje:

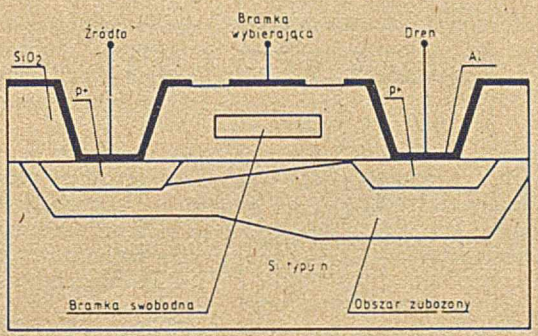
– programowane metodą elektryczną, kasowane metodą nieelektryczną przez naświetlanie promieniami *X* lub promieniami ultrafioletowymi (EPROM);

– programowane i kasowane metodą elektryczną (EEPROM).

PODSTAWOWA KOMÓRKA PAMIĘCI

Podczas budowania pamięci stałych wykorzystuje się zjawisko trwałego magazynowania ładunków w niektórych strukturach MIS (ang. *metal-isolator-semiconductor*).

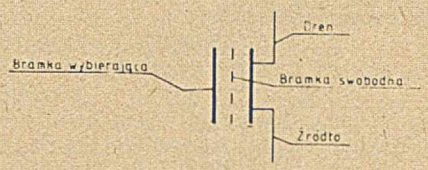
Elementem pamięci stałej jest tranzystor z bramką swobodną (ang. *floating gate avalanche-injector* MOS). Schematyczny przekrój struktury przedstawiono na rys. 1.



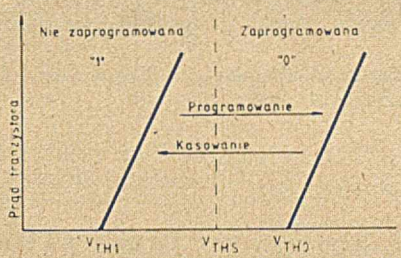
Rys. 1. Schematyczny przekrój struktury

Bramka swobodna z krzemu polikrystalicznego (bez wyprowadzonego kontaktu), ulokowana w odległości ok. 100 nm od powierzchni płytki krzemowej, ładuje się wskutek wstrzykiwania nośników podczas przebicia lawinowego dren–podłoże przy napięciu na drenie powyżej −30 V (obecnie w wielu układach wystarczy −12,5 V).

Pojedyncza komórka pamięci składa się z bramki swobodnej i bramki wybierającej (rys. 2). Bramka wybierająca jest przyłączona do dekodera wierszy, bramka swobodna natomiast jest używana do magazynowania ładunku. Komórka jest programowana przez injekcję elektronów przez tlenek do bramki swobodnej. Zmiana ładunku bramki swobodnej powoduje zmianę progowego napięcia przełączania tranzystora. W stanie początkowym komórka miała niskie napięcie progowe (V_{TH1}), umożliwiające wejście w stan przewodzenia wybranemu bramką tranzystorowi. Programowanie zmienia napięcie progowe na wysokie (V_{TH0}), uniemożliwiające tranzystorowi wejście w stan przewodzenia. Stan komórki można określić badając stan zwrotny progu (V_{THS}), pokazany na rys. 3 (linią przerywaną). Stan naładowania bramki jest bardzo trwały, gdyż w temperaturze 125 °C ładunek maleje do poziomu 70% wartości początkowej, po upływie 10 lat.



Rys. 2.



Rys. 3. Progowe napięcie przełączania

METODY PROGRAMOWANIA PAMIĘCI EPROM

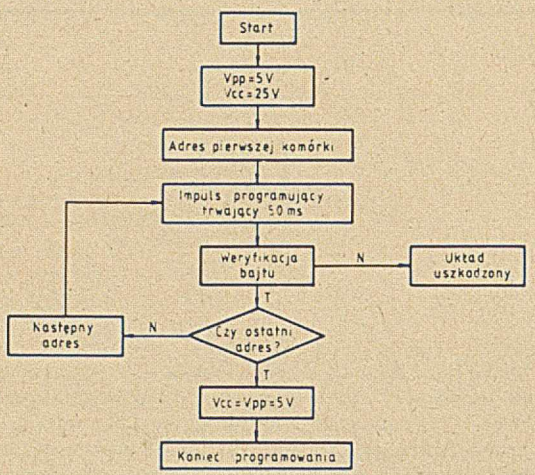
Nie zaprogramowany układ pamięci ma wszystkie bity ustawione w stan wysoki 1; programowanie polega na ustawieniu poszczególnych komórek w stan niski 0. Ustawienie komórki w stan 0 następuje w wyniku podwyższenia progowego napięcia przełączania tranzystora, stanowiącego element pamiętający. Tryb programowania jest włączany w chwili ustawienia napięcia programującego V_{pp} którego wysokość zależy od typu układu. Proces programowania polega na doprowadzaniu krótkiego impulsu programującego na wejście sterujące. W zależności od układu, wejściem sterującym jest wprowadzenie CE lub PGM.

Podczas programowania szczególne znaczenie ze względu na niezawodność układu ma kształt impulsu programującego. Układ pamięci ulega szybkiemu uszkodzeniu w wyniku przerostu napięcia w podawanym impulsie programującym. Z tego właśnie powodu w układach typu 2732, w których podczas programowania jest doprowadzane również napięcie programujące V_{pp} = 25 V (21 V dla 2732A), producent zaleca dołączenie kondensatora blokującego rzędu 0,1 μF między wprowadzenie $\overline{OE}/V_{pp}$ a masę układu.

Metoda programowania standardowego polega na doprowadzaniu do wejścia sterującego PGM impulsu o czasie trwania 50 ms. Taki czas jest z jednej strony wystarczający do naładowania bramki swobodnej, z drugiej strony nie powoduje uszkodzenia tranzystora w wyniku przebicia lawinowego, jakie występuje między bramką a drenem tego tranzystora w procesie programowania. Tolerancja czasu trwania impulsu programującego wynosi 5 ms. Schemat blokowy metody programowania standardowego dla układu 2764 przedstawiono na rys. 4.

Tryb programowania włącza się przez ustawienie napięcia programującego V_{pp} = 25 V (21 V) i napięcia zasilania V_{cc} = 5 V. Po zadaniu

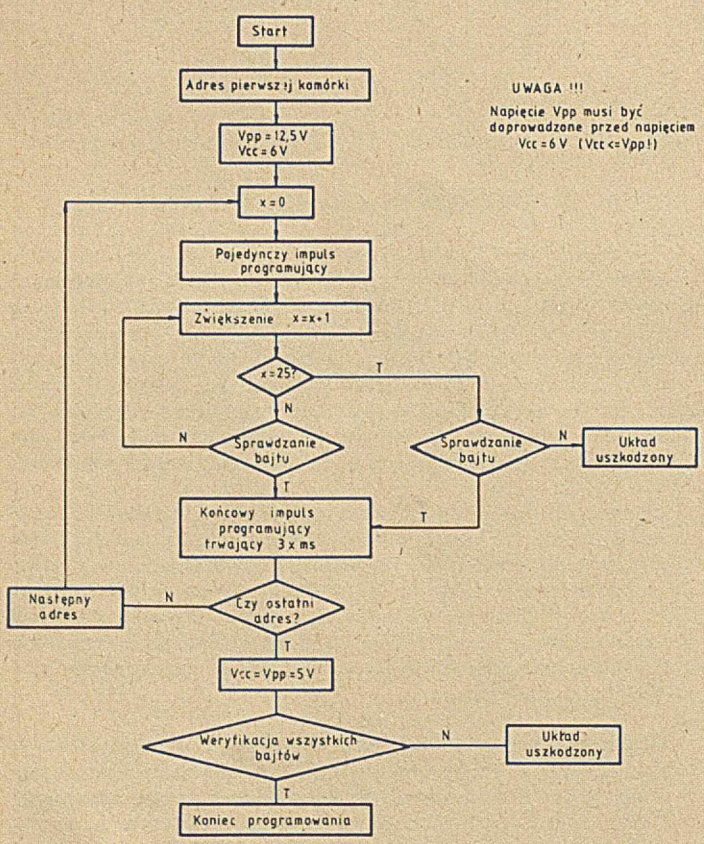
adresu i danych do zapisu, wysyła się impuls programujący o czasie trwania 50 ms. Następnie należy sprawdzić zawartość bajtu. Wynik negatywny praktycznie oznacza uszkodzenie komórki pamięci; jeśli natomiast zawartość komórki była zgodna z wartością programową, to można programować następne komórki. Programowanie kończy się przejściem do trybu spoczynkowego (ang. *standby mode*): $V_{cc} = V_{pp} = 5$ V, $\overline{CE} = 1$



Rys. 4. Schemat blokowy metody programowania standardowego dla układu 2764

Praktycznie w większości pamięci jest możliwe programowanie pojedynczych bajtów, bloków pamięci lub całej zawartości.

Czas programowania układu typu 2716 wynosi 100 s, układu 2732 200 s, a układu 2764 – 400 s.



Rys. 5. Schemat blokowy metody programowania inteligentnego – wersja I

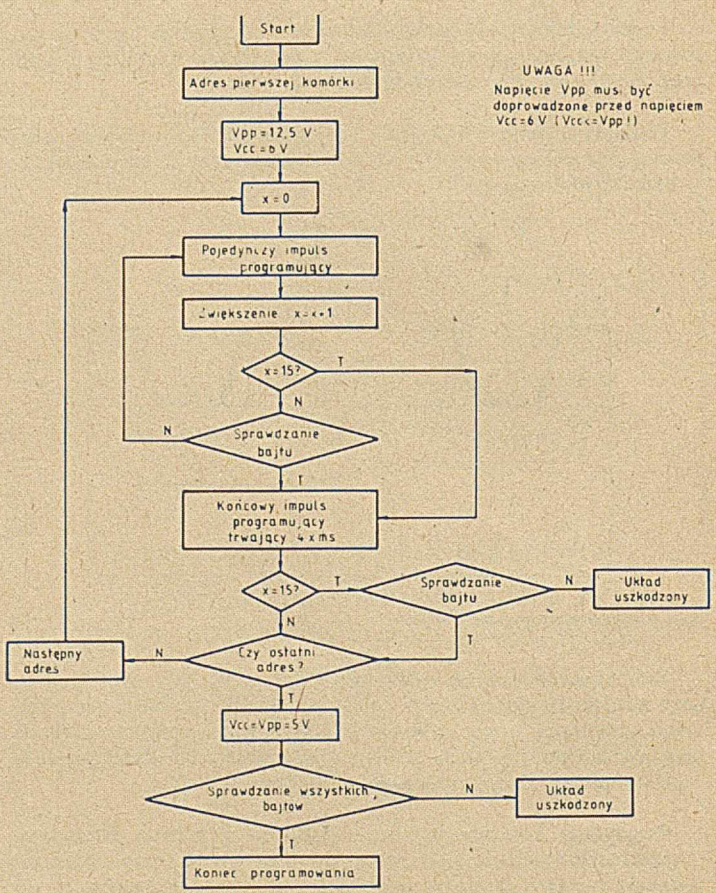
Metoda programowania inteligentnego pojawiła się wraz z pamięciami o pojemności 256 Kb. Czas programowania pamięci o tak dużej pojemności metodą standardową, tj. impulsami o długości 50 ms jest bardzo długi i wynosi ok. 30 minut. Algorytm programowania inteligentnego pozwala natomiast znacznie skrócić czas programowania. Schemat blokowy algorytmu programowania inteligentnego (ang. *intelligent programming algorithm*) dla układu 27256 przedstawiono na rys. 5. Na początku programowania ustawia się napięcia: programujące $V_{pp} = 12{,}5$ V i zasilające $V_{cc} = 6$ V. Po zadaniu adresu i danych do zapisu, doprowadza się impulsy programujące o czasie trwania 1 ms. Po każdym impulsie jest sprawdzana zawartość bajtu; jeśli wynik jest negatywny, to doprowadza się następny, aż do 25. Jeśli wynik sprawdzania po 25 impulsie jest nadal negatywny, to praktycznie oznacza to uszkodzenie komórki pamięci. Przeciętnie jednak 92% wszystkich komórek programuje się po ósmym impulsie.

Stosowanie podwyższonego napięcia zasilania $V_{cc} = 6$ V podczas sprawdzania powoduje podniesienie progowego napięcia przełączania tranzystora będącego elementem pamiętającym komórki. Dlatego wynik sprawdzania wymaga większego marginesu niż w normalnych warunkach. Ten dodatkowy margines, zwiększający pewność prawidłowego odczytu pamięci w trybie normalnym przy $V_{cc} = 5$ V, uzyskuje się przez dodanie końcowego impulsu programującego (ang. *overprogramming impuls*).

Gdy wynik sprawdzania zawartości bajtu jest pozytywny, doprowadza się końcowy impuls programujący. Czas trwania tego impulsu musi być trzy razy dłuższy niż suma czasów wszystkich milisekundowych impulsów doprowadzanych wcześniej, przy programowaniu tego bajtu. Jeśli przykładowo, bajt wymagał tylko jednego impulsu, to końcowy impuls programujący będzie trwał 3 ms, lecz jeśli było potrzebnych maksymalnie 25 impulsów programujących, to czas trwania będzie wynosił 75 ms.

Po zaprogramowaniu wszystkich bajtów ponownie sprawdza się całą zawartość pamięci przy napięciu zasilania $V_{cc} = 5$ V. Dopiero taka kontrola zapewnia niezawodne zaprogramowanie pamięci.

Podobną metodą opartą na przedstawionych zasadach można zastosować do programowania wcześniejszych pamięci, tj. 2764 i 27128, przy napięciu programującym $V_{pp} = 12{,}5$ V. Schemat blokowy algorytmu przedstawiono na rys. 6. Różnica w stosunku do algorytmu z rys. 5 polega na:



Rys. 6. Schemat blokowy metody programowania inteligentnego – wersja II

- ograniczeniu maksymalnej liczby impulsów do 15,
- wydłużeniu końcowego impulsu programującego do 4x.

Jeśli komórka nie zaprogramuje się po piętnastu impulsach, to jest dodawany jeszcze końcowy impuls programujący o czasie trwania 60 ms; gdy wynik sprawdzania jest nadal negatywny, to świadczy to o uszkodzeniu układu.

Według tego samego algorytmu były programowane także układy 2732A przy $V_{pp} = 21$ V oraz 2732 i 2716 przy $V_{pp} = 25$ V. W badanych egzemplarzach czas programowania – na przykład – układu 2716 skrócił się ze 100 do 12 s, układu 2732 – z 200 do 24 s, a układu 2764 – z 400 do 48 s.

Nową tendencją obserwowaną ostatnio w konstrukcjach pamięci stałych, jest wprowadzenie trybu autoidentyfikacji (ang. *intelligent identifier mode* – INTEL, *auto select mode* – HITACHI). Tryb ten umożliwia zidentyfikowanie producenta układu oraz typu pamięci na podstawie odczytanego kodu dwójkowego. Dzięki temu jest możliwe automatyczne dopasowanie programatora do rodzaju układu i wybranie odpowiedniego sposobu programowania. Tryb ten działa w zakresie temperatur 25 ± 5 °C. Aby uaktywnić ten tryb, należy na linię adresową A9 doprowadzić napięcie od 11,5 do 12,5 V. Spowoduje to sekwencyjne pojawienie się na magistrali danych dwóch bajtów informacyjnych przełączanych linią adresową *A0*. Wszystkie pozostałe linie adresowe w tym trybie muszą być w stanie 0.

Przykładowe bajty autoidentyfikacji

Kod \ Nóżka	A0	07	06	05	04	03	02	01	00	Szesnastkowo
Kod producenta	0	1	0	0	0	1	0	0	1	89
Kod układu	1	0	0	0	0	0	1	0	0	04

A_9 : V_H = 12V; ($A1-A8$) & ($A10$ $A15$) = 0; $A0$ = 0 lub 1;

Bajt 0 ($A0 = 0$) reprezentuje kod producenta. Bajt 1 ($A0 = 1$) kod identyfikujący typ pamięci. Bit 7 (07) wszystkich kodów informacyjnych jest bitem parzystości.

Przykładowe bajty autoidentyfikacji dla układu 27256 firmy INTEL przedstawiono w tabeli.

KASOWANIE PAMIĘCI EPROM

Kasowanie komórki pamięci polega na rozładowaniu bramki swobodnej tranzystora, co powoduje obniżenie jego progowego napięcia przełączania i ustawienie komórki w stan 1. Po skasowaniu wszystkie komórki pamięci są w stanie 1.

Jednym ze sposobów kasowania jest naświetlanie promieniami ultrafioletowymi. Do całkowitego skasowania jest wymagana dawka 15 Ws/cm². Taką dawkę można uzyskać przez naświetlanie lampą ultrafioletową o długości fali 2537 Å, o natężeniu 12 mW/cm², przez 15–20 minut. Źródło promieniowania powinno mieć filtr, który eliminowałby fale o innych długościach.

Kasować można również promieniami o długości większej niż 4000Å. Mimo tego, że czas kasowania będzie znacznie dłuższy niż przy źródle 2537 Å, naświetlanie lampą fluorescencyjną lub światłem słonecznym spowoduje także wykasowanie. Niezawodność tej ostatniej metody nie jest zbyt duża.

Aby uniknąć przypadkowego skasowania pamięci po zaprogramowaniu, należy kwarcowe okienko pokryć nieprzezroczystą nalepką.

Na rynku są również dostępne pamięci EPROM typu OTP (ang. *one time programming*) – jednorazowego programowania. Są to układy, które ze względu na sposób programowania zakwalifikowano do pamięci EPROM, lecz z powodu braku okienka kwarcowego (praktycznie nie można skasować ich zawartości) należałoby zaliczyć je do pamięci typu PROM. Duży wybór pamięci typu OTP EPROM oferują firmy SINGNETICS i NEC.

Konferencje

EUROMICRO '90

Szesnasta już, doroczna międzynarodowa konferencja EUROMICRO odbędzie się w tym roku w Amsterdamie (Holandia) w terminie **od 27 do 30 sierpnia.** Hasłem przewodnim tej konferencji jest „Sprzęt i oprogramowanie w inżynierii systemowej" *(Hardware and Software in System Engineering)*. Spośród nadesłanych 125 referatów z 23 krajów Komitet Programowy konferencji wybrał do wygłoszenia 83.

Zgodnie z tradycją program przewiduje podział obrad na cztery podstawowe dziedziny zainteresowań. W ramach tych dziedzin określono na podstawie treści referatów następującą tematykę specjalistyczną:

- **sprzęt komputerowy i architektura:**
  systemy RISC, układy VLSI, tolerancja błędów;
- **oprogramowanie:**
  zagadnienia formalizacji, inżynieria oprogramowania, zagadnienia kompilacji, rozwój oprogramowania równoległego;
- **systemy:**
  projektowanie systemów, przetwarzanie równoległe, systemy real-time;
- **zastosowania**
  przetwarzanie sygnałów, sieci neuronowe, przetwarzanie obrazów, systemy rozproszone.

Konferencję poprzedzi dzień seminaryjny, w ramach którego A. Verschueren, z Politechniki w Eindhoven (Holandia), zaprezentuje temat „Projektowanie obiektowe ASIC", a A. Grambert, z firmy Hewlett Packard (Wielka Brytania) – temat „Inżynieria oprogramowania wspomagana komputerem".

Bliższe informacje na temat konferencji można otrzymać pod adresem: **EUROMICRO – The European Association for Microprocessing and Mickroprogramming, P.O. Box 2346, NL – 7301 EA Apeldoorn. The Netherlands, tel. 31-55-55-73-72.**

ZAKŁAD KOMPUTEROWYCH SYSTEMÓW POMIAROWYCH
ul.Zb. z Bogdańca 4
80-419 Gdańsk
digimer
tel. 419519
teleks 51-22-90

Polecamy usługi w zakresie:
- automatyzacji pomiarów, zbierania i przetwarzania danych pomiarowych w oparciu o komputery PC XT/AT,
- konstrukcji testerów produkcyjnych dla urządzeń elektronicznych cyfrowych i analogowych,
- realizacji interfejsów pomiarowych z oprogramowaniem,
- dostaw sprzętu komputerowego.

Oferujemy wyposażenie PC XT/AT z oprogramowaniem:
* PROGRAMATORY PAMIĘCI (E) EPROM (16kb–2Mb);
* PROGRAMATORY PAMIĘCI BPROM, PAL, PEEL, GAL, EPLD, FPL;
* PROGRAMATORY μP JEDNOUKŁADOWYCH;
* STYMULATORY PAMIĘCI EPROM (16kb–512kb, 8 i 16 bit);
* EMULATORY 8031/32/35/48/51/52 (in circuit);
* TESTERY CYFROWYCH UKŁADÓW SCALONYCH
  – dla TTL, CMOS i RAM,
  – autoidentyfikacja, edytor i debuger wzorca;
* WIELOKANAŁOWE PRZETWORNIKI A/D 12 bit, od 40μs, – z separacją galwaniczną preferowane do zastosowań w przemyśle, medycynie i na stanowiskach badawczych;
* KARTY PROTOTYPOWE PC/XT (dł. 32.5 cm);
* ANALIZATORY STANÓW LOGICZNYCH do 100 MHz;
* KASOWNIKI PAMIĘCI EPROM.

Posiadamy w ciągłej sprzedaży i zapewniamy dostawy po atrakcyjnych cenach pamięci EPROM oraz RAM.
Dla naszych dotychczasowych klientów wysyłamy nową wersję oprogramowania programatora PCPE512 po otrzymaniu dyskietki.

0/5/90

JERZY DAWIDOWSKI
Centrum Obsługi Badań Naukowych i Dydaktyki
Wyższa Szkoła Rolniczo-Pedagogiczna
Siedlce

Ręczne rejestratory danych

Przez rejestrowanie danych rozumie się zapis danych w dowolnej postaci umożliwiającej ich dalsze wykorzystanie. Rejestrowanie automatyczne obejmuje również fazę poprzedzającą, tj. pomiar danych [8] – lub inaczej – ustalanie danych. Artykuł jest poświęcony ręcznym urządzeniom do rejestrowania danych w postaci zapewniającej ich dalsze automatyczne przetwarzanie.

Bezpośrednie rejestrowanie danych (BRD) na maszynowych nośnikach danych (MND), tj. rejestrowanie danych u źródła ich powstawania i bez tworzenia zapisu pośredniego, było w centrum uwagi producentów sprzętu informatycznego od czasu masowych zastosowań komputerów. Korzyści stosowania BRD są następujące:

- zmniejszenie pracochłonności pracy ręcznej,
- wyeliminowanie źródła błędów,
- skrócenie cyklu przetwarzania danych.

Jest zrozumiałe, że skonstruowanie urządzeń do BRD stosowanych stacjonarnie w pomieszczeniach była znacznie łatwiejsza niż urządzeń przenośnych, zasilanych bateryjnie oraz przeznaczonych do pracy w zmiennym środowisku klimatycznym. Powstało wiele specjalizowanych urządzeń stacjonarnych, takich jak zautomatyzowane kasy sklepowe, kasy biletowe, terminale do obsługi bankowej i inne. Często rolę urządzeń do BRD spełniają także wszelkiego rodzaju terminale w systemach wielodostępnych oraz mikrokomputery.

Ręczne (ang. *hand-held*) rejestratory danych[1] (ang. *data collectors, data loggers, data terminals, data capture devices*)[2], mimo już ponad dwudziestoletniej drogi rozwojowej, w Polsce dopiero teraz mają szanse wykorzystania. Urządzenia tego rodzaju znajdują powszechne zastosowanie w badaniach rolniczych, zwłaszcza w dziedzinie produkcji roślinnej, gospodarce leśnej i drzewnej, badaniach leśnych i biologicznych, inwentaryzacji zasobów naturalnych, zarządzaniu inwentarzem, badaniach ruchu drogowego lub budownictwie, a więc w takich dziedzinach działalności gospodarczej, w których rejestrowanie danych musi odbywać się w terenie, z dala od źródeł zasilania w energię elektryczną, często w ruchu i w zmiennych warunkach pogodowych. Warunki użytkowania tych urządzeń wyznaczają wymagania, których zazwyczaj nie spełniają urządzenia konstruowane z myślą o stosowaniu ich stacjonarnie i w pomieszczeniu zazwyczaj nie spełniają.

Rys historyczny

Próby i zastosowania bezpośredniego rejestrowania danych na MND w warunkach polowych były podejmowane już w końcu lat pięćdziesiątych [12]. Próbowano stosować karty dziurkowane ręcznie, tzw. *port-a-punch* [11, 12, 14] oraz karty kreskowe (ang. *mark sensing cards*). W pierwszym przypadku specjalnym przebijakiem, przez otwory w plastikowym szablonie, wytłaczano otwory w standardowej marcie kaszynowej. W drugim – zapis wykonywano na karcie maszynowej specjalnym ołówkiem, nanosząc we właściwych kolumnach kreski, które były odczytywane magnetycznie przez specjalne urządzenie dziurkujące w tych miejscach otwory i tym samym przekształcające kartę w typową kartę perforowaną, czytelną dla urządzenia wejściowego komputera.

Pierwsze modele ręcznych, przenośnych dziurkarek typowych kart perforowanych pojawiły się już na początku lat sześćdziesiątych i były dość długo rozwijane i doskonalone. Jeszcze w roku 1979 amerykańska firma Wright Line oferowała przenośne dziurkarki kart jako urządzenia do zdecentralizowanego zbierania danych. Oferta tego rodzaju urządzeń obejmowała nawet możliwość dziurkowania alfanumerycznego z opisywaczem. Urządzenia te zapewniały rejestrowanie danych na standardowych kartach perforowanych różnych typów.

W końcu lat sześćdziesiątych próbowano również stosować formularze ze znakami optycznymi (ang. *optical mark ferms*). Wczytywanie zapisanych danych wymagało tu jednak specjalnego optycznego czytnika dokumentów [7, 14].

Opisane metody BRD miały wiele wad. Nośniki były mało pojemne i bardzo wrażliwe na wszelkie odchylenia od norm zapisu oraz odczytu, a także niemożliwe do stosowania w dowolnych warunkach pogodowych. Korekta błędnych zapisów była kłopotliwa, a czas rejestrowania dłuższy niż przy tradycyjnym zapisie w notatnikach polowych, a następnie przenoszeniu na maszynowe nośniki danych. I chyba z tych względów metody te nie upowszechniły się.

Inny kierunek rozwoju konstrukcyjnego wiązał się z taśmą magnetyczną. W roku 1964 w leśnictwie testowano pierwszy przenośny numeryczny rejestrator danych na taśmie magnetycznej [9], a udoskonalone jego wersje (w tym alfanumeryczna) wytwarzane przez amerykańską firmę Universal Data Acquisition Company Inc. znalazły praktyczne zastosowania. Tego rodzaju rejestratory stosowano m.in. w badaniach rolniczych we Francji i RFN. Wecke [13] pisze również o przenośnym rejestratorze z taśmą metalową, bardziej odporną na zmienne warunki pogodowe.

W roku 1968 Japończycy [1, 2] podjęli próby na skonstruowanym przez firmę OKI Electric Co. rejestratorem przenośnym w postaci końcówki komputerowej z transmisją bezprzewodową (radiową). Rozwiązanie to miało wiele wad, ale przede wszystkim cała aparatura była zbyt ciężka do noszenia (prawie 5 kg; jeden człowiek nosił na plecach nadajnik i baterie, a drugi prowadził pomiary i rejestrowanie) oraz zbyt kosztowna. Mimo wieloletnich prac [10] zrezygnowano z rozwijania tego rodzaju urządzenia wobec szybkiego rozwoju nowoczesnej technologii elektronicznej.

Współczesne ręczne rejestratory danych to albo urządzenia specjalnie to tego celu przeznaczone, wyposażone w pamięć półprzewodnikową do gromadzenia danych, albo miniaturowe mikrokomputery. Rozwój tych pierwszych zaczął się w drugiej połowie lat siedemdziesiątych. Jedną z pierwszych konstrukcji był rejestrator DATAMYTE firmy General Electric Corporation. Kolejne modele serii 800, 900 i 1000 były dość rozpowszechnione. Jeden z wcześniejszych reprezentantów drugiej grupy, rozwijanej na początku lat osiemdziesiątych, to mikrokomputer EPSON HX-20, powszechnie stosowany do dzisiaj, np. w słynnej rolniczej Rothamstead Experimental Station w Anglii.

Wymagania stawiane współczesnym ręcznym rejestratorom danych

Wymienione poniżej wymagania dotyczą urządzeń stosowanych w doświadczalnictwie polowym. Wynikają one z warunków ich użytko-

[1] Termin *ręczny rejestrator danych* przyjęto dla urządzenia do rejestrowania danych trzymanego w ręku.

[2] W języku angielskim dla omawianych urządzeń używa się czasem określenia nie wystarczająco precyzyjnego – *portable* (przenośne) zamiast *hand-held* (ręczne). W informatyce pierwszego określenia używa się również dla urządzeń niekoniecznie o wymiarach i ciężarze pozwalającym na ich trzymanie w ręku i niekoniecznie zasilanych bateryjnie. Drugi termin jest bardziej odpowiedni dla omawianej klasy urządzeń; przyjął się dla urządzeń przenośnych, zasilanych bateryjnie, takich, które rzeczywiście można trzymać w ręku w trakcie wykonywania operacji rejestracji danych i z którymi można swobodnie się poruszać.

Z uwagi na zainteresowania autora zastosowaniami tych urządzeń w doświadczalnictwie polowym (dziale badań agronomicznych) omawia się je na przykładzie wymienionych zastosowań. Nie umniejsza to jednak istotnie ogólnej użyteczności zawartych w artykule informacji.

wania, występujących również w innych dziedzinach zastosowań. Są to:
- praca w zmiennych warunkach pogodowych (deszcz, kurz, silne nasłonecznienie, zmienna temperatura i wilgotność),
- obsługa przez personel nie wyspecjalizowany w rejestracji danych,
- praca przez 8 i więcej godzin dziennie,
- wykonywanie kolejnych pomiarów w warunkach przemieszczania się z urządzeniem rejestrującym z jednego miejsca w inne,
- rejestrowanie różnorodnych sekwencji pomiarów w kolejnych dniach, a nierzadko w ciągu dnia,
- rejestrowanie danych alfanumerycznych,
- konieczność możliwie szybkiego wykonywania pomiarów i rejestracji.

Z powyższych warunków stosowania wynikają następujące wymagania:
- odporność na znaczne wahania temperatur, kurz, dużą wilgotność, deszcz,
- czytelność wyświetlacza w promieniach słonecznych,
- trwała konstrukcja,
- mały ciężar,
- łatwość obsługi,
- alfanumeryczna klawiatura,
- programowalność procedur rejestrowania,
- zasilanie bateryjne,
- możliwość wykonywania operacji w dostatecznie długim czasie, między powtórnym ładowaniem lub wymianą baterii,
- możliwość szybkiego rejestrowania danych,
- łatwość poprawiania błędnie zapisanych danych, a więc kasowania, zamiany i wstawiania,
- możliwość przeglądania zapisanych już danych.

Przegląd współczesnych ręcznych rejestratorów danych

W tabeli scharakteryzowano 10 modeli rejestratorów pod względem ich parametrów technicznych i użytkowych[3]. Modele te są od kilku lat stosowane m.in. w praktyce doświadczalnictwa polowego USA, Japonii i wielu krajów Europy Zachodniej. Zapewniają one elastyczność organizacji wprowadzania danych, wynikającą albo z możliwości programowania w znanych językach programowania (większość rejestratorów) lub w specjalnie zaprojektowanym do celów rejestrowania danych języku Polycode, albo też z możliwości adaptacji przez wprowadzenie programu rejestrowania z mikrokomputera, na którym program przygotowano (MICRONIC 203, DATAMYTE 1003), lub zmianę pamięci EPROM z takim programem (MSI/85).

Dane zarejestrowane w pamięci urządzenia na przykład po jednodniowej sesji można przesłać do dalszego przetwarzania przez odpowiedni interfejs bezpośrednio do komputera lokalnego lub przez modem akustyczny i sieć telefoniczną do komputera odległego.

Aktualne ceny omawianych urządzeń nie są znane autorowi. Warto jednak podać ceny z lat 1984–1985. Były one dość wysokie; wynosiły od 600 dolarów za MICRONIC 203 do 3500 za HUSKY HUNTER i DATAMYTE.

* * *

W pracach [5, 6, 10], zawierających oceny porównawcze niektórych rejestratorów danych, można znaleźć całkowicie przeciwstawne opinie. Jedni uważają, że rejestratory HUSKY HUNTER, EPSON HX-20 lub DATAMYTE są zbyt ciężkie, inni z powodzeniem je stosują. Celem artykułu nie jest jednak ocena tych urządzeń, lecz zwrócenie uwagi na obszerną już klasę urządzeń, które w krajach zachodnich zyskują coraz powszechniejsze zastosowanie.

W Polsce wskutek dotarcie mikrokomputerów do mniejszych jednostek działalności gospodarczej lub badawczej, takich jak np. różnego rodzaju rolnicze zakłady doświadczalne, zastosowanie rejestratorów jest obecnie uzasadnione. Można też przypuszczać, że obecne ceny rejestratorów danych będą również bardziej zachęcające.

[3] Informacje w tabeli przygotowano na podstawie prac [3, 5, 6, 10] oraz prospektów rejestratorów MICROFIN, DATAMYTE, HUSKY HUNTER, EPSON HX-20, MSI/85 i POLYCORDER.

Charakterystyka ręcznych rejestratorów danych

Parametr \ Model producent, kraj	MICRONIC M445L, ?, Szwecja	MICRONIC 203, ?, Szwecja	MICROFIN, Microfin Systems Ltd. Anglia	HUSKY HUNTER, Husky Computer Ltd. Anglia	EPSON HX-20 Epson (UK) Ltd., Anglia	PHC-800 Sanyo, Japonia	RL-H 1000 Matsushita Japonia	DATAMYTE 1003, EGC, USA	MSI/85, MSI, USA	POLY-CORDER 600 Series, Omnidata Int. Inc., USA
Typ urządzenia	rejestrator	rejestrator	rejestrator	mikro-komputer	mikro-komputer	mikro-komputer	mikro-komputer	rejestrator	rejestrator	rejestrator
Ciężar (kg)	0,45	0,15	0,9	1,15	1,7	0,4	0,6	1,5	0,45	0,8
Wymiary (cm)	17×9×4	14×7×2	25×11×5	22×17×3	29×22×4	21×10×3	23×10×3	25×33×4	15×9×4	20×10×4
Wyświetlacz:										
liczba znaków w wierszu	16	12	16 16	32	20	?	?	16	16	16
liczba wierszy	1	1	1 2	8	4	?	?	1	1	4
LED	–	–	+	–	–	–	–	–	–	–
LCD	+	+	+	+	+	+	+	+	+	+
alfanumeryczny	–	–	+	+	+	+	+	+	+	+
graficzny	–	–	–	+	+	–	–	–	–	–
Język programowania	asembler Pascal	adaptowalny	adaptowalny	Pascal, Fortran, Basic	Basic	Basic	Snap Basic	adaptowalny	adaptowalny	Polycode
System operacyjny	–	–	–	CP/M	–	–	–	–	–	–
Pojemność pamięci (KB)	–	16	32	–	–	4–22	4–8	–	–	64–256
ROM	6	–	–	48	32	–	–	–	8–16	–
RAM	12–24	–	–	88–264	16	–	–	64	4–32	–
Kod danych w pamięci	BCD	ASCII	BCD, ASCII	ASCII, HEX	ASCII	?	?	ASCII	?	?
Podział pamięci	–	–	–	–	+	?	?	–	?	?
Urządzenia wbudowane:										
pamięć kasetowa	–	–	–	–	+	–	–	–	–	–
drukarka	–	–	–	–	+	–	–	–	–	–
interfejs RS 232	+	+	+	+	+	+	+	+	+	+
interfejs z komputerem	IBM 370 IBM 3081 CP/M Micros	POP-11/44	CTL 8066 CBM PET APPLE II SIRIUS	IBM PC SOLAR 16/85 CROMENCO SIRIUS	PRIME 550 VAX 750 IBM PC CROMENCO SIRIUS	?	?	CROMENCO	?	?
Baterie										
możliwość ładowania	+	–	+	+	+	+	+	+	+	+
okres podtrzymania pamięci (godz.)	2000	2000	?	50	?	?	?	600	?	5 lat
Urządzenia zewnętrzne										
czytnik kodu kreskowego	+	+	+	+	+	?	?	+	+	+
drukarka	+		+	+	+	?	+	–	–	+
Odporność na pogodę	+ (z osłoną)		+	+	+	?	?	+	?	+

Zainteresowanym stosowaniem ręcznych rejestratorów danych można zalecić następujący tok postępowania prowadzący do trafnego użytkowo i ekonomicznie wyboru urządzenia:

- precyzyjne określenie warunków użytkowania rejestratora,
- precyzyjne określenie parametrów technicznych i wymagań użytkowych wynikających z warunków zastosowania,
- wybór kilku modeli spełniających wymagania z dostatecznie szerokiej oferty (autor artykułu sądzi, że obecna oferta światowa może liczyć co najmniej 50 różnych modeli),
- testowanie wybranych modeli w praktyce,
- wybór modelu najlepiej ocenionego w praktyce i zadowalającego cenowo,
- rozpoczęcie realizacji oprogramowania do rejestrowania danych i ewentualnie sprzęgu programowego z istniejącym bądź realizowanym komputerowym systemem przetwarzania danych,
- zakup niezbędnej liczby urządzeń wybranego modelu i ich wdrożenie do konkretnych zadań rejestrowania danych.

Wydaje się, że w dalszym ciągu równoległy rozwój obu koncepcji urządzeń, tj. urządzenia specjalizowanego jako rejestrator danych i ręcznego mikrokomputera, będzie utrzymany, przy czym możliwości użytkowe i parametry techniczne przedstawicieli obu grup mogą być dość bliskie. Dalsze postępy w rozwoju technologii elektronicznej oraz informatycznej powinny również wskrzesić koncepcję ręcznych inteligentnych terminali i transmisji bezprzewodowej (niekoniecznie radiowej, ponieważ były już rozwiązania z transmisją na promieniach podczerwonych), co niewątpliwie zwiększy krąg zastosowań i możliwości funkcjonalne omawianej klasy urządzeń.

LITERATURA

[1] Akemine H., Kawai T., Okabe S.: An Installation for Automatic Data Transfer, Record and Process in Agricultural Field Experiments. Jap. Agr. Res. Quart., No. 5 (2), pp. 67 72, 1970
[2] Akemine H., Kawai T., Kumagai K., Okabe S., Suzuki S., Watanabe T.: An Improved System for Data Transfer, Record and Processing, Assembled with Automatic Balance and Scaler. Proc. of the 4th Int. Conf. on the Mech. of Field Exper., Iowa State University Ames, pp. 277 284, 1976
[3] Anonim: Technical Working Party on Automation and Computer Programs-Report of 3rd Session, Wageningen Netherlands. UPOV Geneva, document TWC/III/13, 1985
[4] Haufe W.: Probleme der Planung und Vorbereitung sovie der rationellen Datenerfasung bei der Durchfuhrung von Feldversuchen. EDV in Med. und Biol. No. 1, 1971
[5] Law J. R., Reeves J. C.: The Use of Hand-held Terminals in National Institute of Agricultural Botany Field Trial Recording. J. natn. Inst. Agric. Bot., No. 16, pp. 397 413, 1984
[6] Leuchovius T.: A Computerized System for Data Collection in the Field. Proc. of the 7th Int. Conf. on the Mech. of Field Exper. pp. 270 274, Debrecen Hungary, 1988
[7] Rowell J. G.: A Rapid Method of Data Collection and Processing. J. Agric. Sci., No. 77, pp. 379 384, 1971
[8] Rymaszewski R.: Pomiar i automatyzacja. Pomiary, Automatyka, Kontrola, nr 2, s. 65 66, 1973
[9] Stewart D. H.: Portable Magnetic-tape Encoders for Computer Data. Appl. Stat., No. 18 (1), pp. 98 102, 1969
[10] Suzuki S., Yamaguchi H.: Efficient Use of Electronic Notebook in Breeding Experiments. Proc. of the 6th Inf. Conf. on Mech. of Field Exper., pp. 401 407, Dublin, Ireland, 1984
[11] Thorne C.: Recording Numerical Data on port-a-punch cards. Appl. Stat. No. 18 (1), pp. 88 89, 1969
[12] Twersky M.: Recording Results „On-the-spot" Using Manual Punch Card Equipment. Agr. J. No. 52, pp. 121 122, 1960
[13] Wecke W.: EDP (EDV) Compatible Collection of Measuring Data. Proc. of the 4th Inf. Conf. on the Mech. of Field Exper., pp. 271 276, Iowa State University Ames
[14] Wuppermann D.: Untersuchungen über die Eignung verschiedener Datenträger bei Benutzung durch ungeübte Personen. EDV in. Med. und Biol. No. 1, pp. 11 14, 1972.

Nowe książki

WYDAWNICTWA NAUKOWO-TECHNICZNE

Jerzy Szczepkowicz: Turbo Pascal 4.0 z dodatkiem wersji 5.0. Warszawa 1989, nakład 20 tys. egz., s. 320, ISBN 83-204-1208-0

Książka zawiera obszerny opis systemu programowania Turbo Pascala 4.0 konkretnej realizacji języka programowania Pascal, opracowanej dla rodziny komputerów typu IBM PC w firmie Borland International Inc. Opisano w niej wszystkie zmiany, ograniczenia i rozszerzenia języka wzorcowego, obiekty standardowe oraz szczegóły eksploatacji głównych elementów systemu, m.in. komplet dyrektyw i zleceń. Zwrócono uwagę na operacje językowe, zarządzanie pamięcią, programowanie sterowników ekranu w trybach znakowych i graficznych oraz zasady współpracy programu z systemem operacyjnym. Omówiono zasady programowania modularnego. Podano wskazówki dotyczące kłopotliwych aspektów programowania oraz przykłady podprogramów, modułów i programów, zawierające praktyczne informacje o działaniu urządzeń komputera. Prawie wszystkie wiadomości o Turbo Pascalu 4.0 dotyczą również wersji 5.0, najważniejsze informacje o wersji 5.0 podano w oddzielnym dodatku.

Książka jest przeznaczona dla szerokiego kręgu odbiorców i może służyć zarówno jako podręcznik programowania w Turbo Pascalu 4.0 i 5.0, jak i poradnik zawodowego programisty, operatora i użytkownika komputera typu IBM PC.

A. Nico Habermann, Dewayne E. Perry: Ada dla zaawansowanych. Warszawa 1989, nakład 10 tys. egz., s. 464, ISBN 83-204-1058-4

Książka stanowi monografię języka Ada będącego jednym z najnowocześniejszych języków programowania. Podano w niej ogólną charakterystykę języków programowania, uwypuklając właściwości Ady stanowiące odbicie nowoczesnych tendencji w inżynierii oprogramowania. Omówiono konstrukcje sekwencyjne, koncentrujące się na typach danych w Pascalu i abstrakcji danych w Adzie. Przedstawiono zagadnienia współbieżności, a więc synchronizacji, wzajemnego wykluczania i komunikacji.

W każdym z rozdziałów książki omawia się pewien typowy problem. Budując jego rozwiązanie w Adzie, wprowadza się w miarę potrzeby stopniowo wszystkie pojęcia tego języka. Pokazanie analogicznego rozwiązania w Pascalu umożliwia porównanie obu języków. Rozdziały uzupełniono ćwiczeniami do samodzielnego wykonania.

Książka jest przeznaczona dla programistów, projektantów systemów przetwarzania informacji, pracowników nauki zajmujących się informatyką oraz dla studentów kierunków informatycznych.

Jacek Błażewicz: Złożoność obliczeniowa problemów kombinatorycznych. Warszawa 1988, nakład 3000 egz., s. 196, ISBN 83-204-0992-6

W książce przedstawiono zarys teorii złożoności obliczeniowej problemów kombinatorycznych, rozpoczynając od problemów decyzyjnych i przechodząc do problemów optymalizacyjnych. Omówiono także metodykę rozwiązywania trudnych problemów optymalizacji kombinatorycznej. Zwrócono uwagę na złożoność czasową i pamięciową problemów i na wpływ równoległości obliczeń na złożoność. Rozważania teoretyczne zilustrowano przykładami omawiającymi zastosowanie wprowadzonych pojęć i metody analizy w konkretnych sytuacjach praktycznych.

Książka jest przeznaczona dla programistów, projektantów systemów przetwarzania informacji, pracowników nauki zajmujących się informatyką oraz dla studentów kierunków informatycznych, technicznych i ekonomicznych.

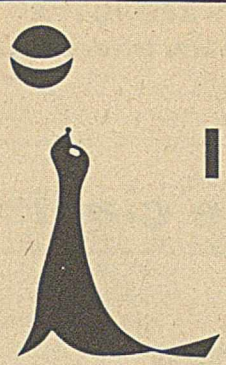
M. Ben-Ari: Podstawy programowania współbieżnego. Warszawa 1989, nakład 6000 egz., s. 148, ISBN 83-204-1121-1

Książka stanowi zwięzły wykład zagadnień współbieżności. Po kilku zadaniach służących do porównania programowania sekwencyjnego ze współbieżnym omówiono podstawowy problem synchronizacyjny – zasadę wzajemnego wykluczania, a także najważniejsze mechanizmy programowania współbieżnego: semafory, monitory oraz mechanizm spotkania (*rendez-vous*) użyty w języku Ada. Opisano również problem „pięciu filozofów".

Książka jest przeznaczona dla programistów, projektantów systemów przetwarzania informacji, pracowników nauki zajmujących się informatyką oraz dla studentów kierunków informatycznych.

Teodor Winkler: Komputerowy zapis konstrukcji. Warszawa 1989, nakład 4000 egz., s. 136, ISBN 83-204-1005-3

Książka dotyczy problematyki wykorzystania nowoczesnych pomocy inżynierskich, takich jak komputery i automatyczne maszyny kreślarskie, w celu zracjonalizowania prac w procesie projektowo-konstrukcyjnym. Jest przeznaczona dla studentów, konstruktorów w zakładach przemysłowych oraz pracowników biur konstrukcyjnych, którzy chcą poznać najnowocześniejsze urządzenia komputerowe i metody stosowane podczas procesu projektowo-konstrukcyjnego.



INTERSOFT Sp. z o.o.

Nasze Biuro Handlu Zagranicznego
poleca swoje usługi
w zakresie exportu-importu każdego towaru.
Prowadzi sprzedaż
profesjonalnego sprzętu komputerowego i peryferii.

1. JĘZYKI PROGRAMOWANIA, BIBLIOTEKI, GRAFIKA	
Basic – programowanie w języku Basic cz. III	52 000 zł
Turbo Basic	183 000 zł
Proffesional Fortran + dodatki	144 000 zł
Fortran 77	132 600 zł
Fortran 80	65 300 zł
Język C – zastosowanie dla zaawansowanych	214 200 zł
Kompilator języka C	106 000 zł
System Turbo C V.1.5	79 000 zł
Turbo C biblioteka	150 000 zł
Turbo C podręcznik użytkownika	109 000 zł
Turbo Pascal	123 500 zł
Turbo Pascal V.4.0	277 000 zł
Turbo Pascal V.5.0	161 000 zł
Turbo Pascal V.5.5 dodatek do T.P.V.5.0	56 000 zł
Turbo Power Tools Plus	124 000 zł
Turbo Database Toolbox	33 000 zł
Modula 2 86	125 000 zł
Turbo Pascal 5.0 opis języka	189 000 zł
Turbo Pascal 4.0 (aut. Jan Bielecki)	153 000 zł
2. BAZY DANYCH, PAKIETY ZINTEGROWANE	
Clipper do dBase III	30 000 zł
Clipper do dBase III +	131 000 zł
Clipper 86	158 000 zł
Clipper 87	212 700 zł
Clipper 87 do dBase III +	214 000 zł
dBase II i III – podręcznik dla zaawansowanych	205 000 zł
dBase III + instalacja pakietu relacyjnej bazy danych	22 000 zł
dBase III + zastosowanie	205 000 zł
dBase III + opis pakietu sieciowego	55 000 zł
dBase III + programowanie	126 000 zł
dBase III + poznawanie	111 000 zł
dBase III – poradnik encyklopedyczny	237 000 zł
dBase IV	249 000 zł
Framework II P	162 000 zł
Informix	261 000 zł
Lotus 1–2–3 + dodatek (komplet)	113 000 zł
Multiplan	66 000 zł
Pakiet C-ISAM	34 700 zł
CX-Fortran kompilator języka do systemu DOS 2	114 300 zł
Plib 86/Plink 86 opis języka	47 0000 zł
Statgraphics	91 000 zł
Programowanie systemów baz danych języka Clipper	107 000 zł
Kompilator CBasic tom. II	92 000 zł
3. SYSTEMY OPERACYJNE, PROGRAMY UŻYTKOWE	
System operacyjny DOS cz. 1 i 2	155 000 zł
System operacyjny DOS 3.10	47 00 zł
System operacyjny DOS 3.20	181 000 zł
System operacyjny DOS 3.30	180 000 zł
System operacyjny DOS 4.0	202 000 zł
Eureka	64 000 zł
Norton Commander	63 000 zł
Polywindows	50 000 zł
Sidekick	44 000 zł
4. EDYTORY	
Chl-2 Writer	31 000 zł
Personalny Edytor PE	68 300 zł
WordStar 2000	142 000 zł
5. CAD – SIECI	
AutoCad 2.17	137 000 zł
AutoCad	30 600 zł
Lanlink 4.0 sieć lokalna	80 000 zł
NOWELL, instalacja sieci Arcnet	9 000 zł
NOWELL, instalacja sieci	65 000 zł
NOWELL, podręcznik użytkowania sieci	116 000 zł
Or-Cad	134 000 zł
6. ROZMAITOŚCI	
Przewodnik programisty	205 000 zł
Printer STAR NL-10 instrukcja	20 000 zł
Wprowadzenie do użytkowania komputerów osobistych klasy IBM PC	43 000 zł
Drukarka NX-16 instrukcja	106 000 zł
Drukarz	14 000 zł
Karta – system rozliczania kart drogowych	10 000 zł
Appsgen	18 000 zł

Zamówienia prosimy kierować pod adresem:
INTERSOFT Sp. z o.o.
00–496 Warszawa, ul. Krucza 16/22
telefony: 28-44-81 w. 284, 352; 28-29-53
teleks: 812414

EO/232/89

MIECZYSŁAW KŁOPOTEK
Instytut Podstaw Informatyki PAN

Rozpoznawanie słów w mowie ciągłej

Rozpoznawanie słów w mowie ciągłej jest częścią procesu rozpoznawania mowy, która ma zadanie wykrycia występowania i umiejscowienia w sygnale mowy wybranych słów (zadanych przez słownik). Rozpoznawanie słów jest istotnym aspektem automatycznego rozumienia mowy. Stanowi fazę przejściową między akustyczno-fonetycznym przetwarzaniem sygnału mowy [9, 10] a analizą znaczenia.

Rozpoznawanie słów w mowie ciągłej znajduje zastosowanie m.in. w następujących dziedzinach:
automatyczny system informacji telefonicznej (np. [7, 8]),
system ekspertowy na sali operacyjnej (chirurg nie może dotykać klawiatury podczas operacji),
wykrywanie słów (haseł) w automatycznym nasłuchu (zastosowanie wojskowe) [3].

Główna trudność automatycznego rozpoznawania słów wynika z małej rozpoznawalności zgłosek, spowodowanej koniecznością uwzględnienia wielu mówiących osób oraz ograniczeniami wynikającymi np. z zawężenia pasma mowy w sieci telefonicznej. W pracy [11] podaje się, że w wyniku analizy segmentu sygnału mowy system rozpoznawania zgłosek generuje tylko w 51% przypadków poprawną zgłoskę jako pierwszą alternatywę, a tylko w 79% przypadków poprawna zgłoska znajduje się wśród pięciu pierwszych alternatyw. Przyczyną jest tu trudność ustalenia granic zgłosek w sygnale mowy, wskutek czego niektóre segmenty zgłosek zlewają się, a inne ulegają podziałowi.

Zadaniem rozpoznawania słów jest ograniczenie eksplozji hipotez zgłoskowych, przy zastosowaniu słownika rozpoznawanych słów, mając na celu utrzymanie złożoności analizy na wyższych poziomach systemu rozpoznawania mowy (analiza syntaktyczna, semantyczna, pragmatyczna) w rozsądnych granicach.

W artykule przedstawiono wybrane algorytmy rozpoznawania słów oraz opisano ogólny schemat procesu rozpoznawania słów w mowie ciągłej. Prezentowana metodologia znalazła zastosowanie w znanych systemach rozpoznawania mowy:
- HEARSAY II [2]
- HWIM [5]
- EVAR [11] i inne.

ETAPY PROCESU ROZPOZNAWANIA SŁÓW

Rozpoznawanie słów można podzielić na następujące etapy:
- ustalenie pozycji końca grupy słów,
- określenie pozycji początkowej,
- generowanie hipotez grup słów,
- przypisanie hipotez słów,
- ocena hipotez,
- eliminacja konkurujących hipotez,
- redukcja liczby wygenerowanych hipotez.

W celu ustalenia pozycji końca słów ze słownika ocenia się dla każdego słowa „odległość" kolejnych fragmentów sygnału mowy od wzorca fonetycznego. Jeżeli wartość odległości (lub miary odwrotnej, czyli podobieństwa) przyporządkować końcowym segmentom porównywanych fragmentów sygnału mowy, to otrzymuje się funkcję prawie ciągłą, której minima (maksima dla podobieństwa) są uważane za najprawdopodobniejsze miejsca zakończenia danego słowa w sygnale mowy.

Ponieważ w języku polskim (i innych językach) istnieją słowa identyczne fonetycznie (np. *„buk"*, *„Bug"*, *„bóg"* i *„Bóg"*, *„lut"*, *„lód"* i *„lud"*), można połączyć je w grupy słów, dla których odległość fonetyczną mierzy się tylko raz.

Określiwszy pozycję końca (pozycje końców) grupy można ustalić też segment początku grupy w sygnale mowy, by określić później, jakie hipotezy słów są wzajemnie sprzeczne (pokrywają częściowo ten sam fragment sygnału mowy). W ten sposób każdemu słowu przyporządkowuje się piątkę postaci:

(*identyfikator słowa, identyfikator grupy słów, pozycja początku, pozycja końca, podobieństwo fonetyczne*).

Jako ocenę hipotezy można przyjąć tzw. podobieństwo między wzorcem słownika a wycinkiem sygnału mowy. Podane dalej miary podobieństwa zależą jednak od długości słowa ze słownika i nie mogą być podstawą porównania wiarygodności hipotez słów różnej długości. Dlatego stosuje się transformację:

$$\textit{ocena hipotezy} = -\log(\textit{podobieństwo wzorzec–sygnał}) / \textit{długość wzorca}.$$

Jeżeli w słowniku fonetycznym znajdą się – oprócz wymowy standardowej słowa – także dialektowe warianty wymowy (np. „kelner" i „kielner"), to podczas generowania hipotez pojawi się nieprzyjemny efekt częstego występowania hipotez należących do tego samego słowa (będących jego wariantami wymowy) i nakładających się w dużym stopniu na siebie. W takich wypadkach eliminuje się hipotezy identycznych słów pozostawiając jedną o największej wartości funkcji oceny.

W ostatnim etapie rozpoznawania słów należy ograniczyć liczbę generowanych hipotez słów, np. do 10 hipotez na jedno słowo ze słownika (10 najwyżej ocenionych hipotez). W ten sposób wycinkowi sygnału mowy przyporządkowuje się hipotezy słów prawdopodobnie występujących w danym wycinku, które potem są przetwarzane w module nadrzędnym. Korzyść wynikająca z generowania hipotez jest następująca: aby rozpoznać 80% występujących w tekście słów, moduł generowania hipotez słów generuje 25 hipotez słów na segment mowy, podczas gdy na poziomie rozpoznawania zgłosek potencjalna liczba słów na segment wynosi co najmniej 500, przy tym samym stopniu wykrywania.

PORÓWNYWANIE WZORCA SŁOWA Z WYCINKIEM SYGNAŁU MOWY – ODSTĘP I PODOBIEŃSTWO

Możliwe są co najmniej trzy rodzaje pomiaru odstępu między wzorcem słowa a wycinkiem sygnału mowy:
- syntaktyczny,
- geometryczny,
- statystyczny.

Odległość syntaktyczna jest mierzona ciągiem znakowym reprezentującym wzorzec słowa i ciągiem znaków reprezentującym ciąg segmentów sygnału mowy. Oblicza się tzw. *odstęp Levenshteina* [6], będący minimalną (ważoną) liczbą operacji transformujących jeden ciąg znakowy w drugi.

Dopuszcza się następujące operacje:
- błędne rozpoznanie litery, np.

wzorzec	wycinek sygnału
goniec	*koniec*

- „połknięcie litery, np.

walec	*walc*

- „wstawienie" litery

szkło	*szkoło*

Klasyczny odstęp Levenshteina nadaje każdej operacji jednakową wagę (1), można jednak większą wagę przypisać mniej wiarygodnym

błędom (np. „połknięcie" może być mniej wiarygodne niż błędne rozpoznanie), uzyskując lepsze dopasowanie do rzeczywistości.

Niewątpliwą wadą tej miary odległości jest to, iż identycznie ocenia się fonetycznie uzasadnione (np. *goniec* i *koniec*) i nieuzasadnione (np. *soja* i *sowa*) różnice między wzorcem a porównywanym sygnałem mowy.

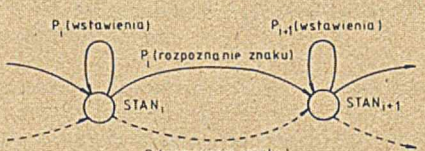
Metoda geometryczna odwołuje się bezpośrednio do sygnału mowy, pomijając krok rozpoznawania zgłosek. Wzorcową i rzeczywistą krzywą sygnału porównuje się przez „rozciągnięcie" obu do standardowej długości (rozciągnięcie niekoniecznie jest równomierne) i przez (ważone funkcją rozciągania) całkowanie różnicy „rozciągniętych" krzywych. W praktyce oczywiście nie przechowuje się i nie porównuje krzywych ciągłych, lecz dyskretne wartości próbkowe. Obowiązuje jednak zasada, że częstość próbkowania musi dwukrotnie przekraczać częstotliwość sygnału próbkowanego.

Wadą tej metody mierzenia odstępu jest nie tylko pracochłonność procesu obliczeniowego, lecz także uzależnienie wzorca od osoby mówiącej.

Podejście statystyczne jest modyfikacją podejścia syntaktycznego, polegającą na uwzględnieniu prawdopodobieństw odchyleń fonetycznego zapisu wzorca od wyniku rozpoznania analizatora zgłoskowego sygnału mowy. Zastosowanie metody statystycznej jest jednak bardziej złożone i wymaga kilku operacji przygotowawczych:
- dokonania treningowego nagrania sygnału mowy (o wystarczającej częstości występowania poszczególnych zgłosek),
- ręcznego podzielenia na zgłoski i porównania z wynikami analizatora zgłoskowego, w celu ocenienia prawdopodobieństwa błędnego rozpoznania poszczególnych zgłosek,
- przeprowadzenia obliczeń optymalizacyjnych macierzy błędnego rozpoznania, w celu jej lepszego dopasowania do tekstu nagranego jako całość, a nie tylko poszczególnych zgłosek.

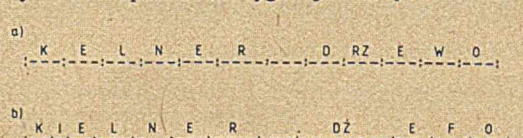
Przed opisaniem procedury uczenia systemu, warto przedstawić sposób mierzenia odstępu przy założeniu, że proces treningu został już zakończony. Za miarę podobieństwa między wzorcem W słowa w słowniku a rozpatrywanym przez analizator wycinkiem F sygnału wejściowego można uznać prawdopodobieństwo warunkowe $P(F/W)$ zinterpretowania przez analizator zgłoskowy sygnału wejściowego jako ciąg fonemów F, jeżeli faktycznie na wejściu pojawił się sygnał odpowiadający ciągowi fonemów W.

Prawdopodobieństwo to oblicza się na podstawie modelu Markowa wzorca słowa. Model Markowa pojedynczego słowa przedstawiono fragmentarycznie na rys. 1.



Fot. 1. Fragment modelu pojedynczego słowa

Model ten wynika z następującego rozumienia procesów rozpoznawania zgłosek. Niech będzie dany wzorcowy sygnał wejściowy jak na rys. 2a, gdzie pionowe kreski dzielą sygnał prawidłowo na odcinki odpowiadające fonemom (zgłoskom). Na skutek różnych przypadkowych wpływów (a także natury mikrofonemicznej mowy [4]) analizator zgłoskowy może rozpoznać ten sygnał jak na rysunku 2b.



Fot. 2. Poprawna (a) i błędna (b) interpretacja fonetyczna sygnału wejściowego – możliwe błędy interpretacji

Mogą wystąpić następujące błędy analizatora:
- błędne rozpoznanie zgłoski wejściowej,
- zlanie kolejnych zgłosek (zatarcie granicy między zgłoskami),
- rozszczepienie zgłoski (wykrycie granicy między zgłoskami, która nie istnieje naprawdę).

Przez model słowa można w świetle powyższego rozumieć graf, którego węzły – zwane stanami – odpowiadają „prawdziwym" zgłoskom na wejściu, a krawędzie skierowane są znakowane wykrywanymi przez analizator zgłoskami i łączą stany odpowiadające „prawdziwym" segmentom zgłoskowym, przez które przechodzi głowica analizatora zgłoskowego.

W danym kroku rozpoznawania zgłoski mogą wystąpić zdarzenia:
- głowica pozostała na tej samej faktycznej zgłosce, co oznacza w praktyce rozszczepienie zgłoski (pętla w grafie modelu słowa),
- głowica przeskoczyła jedną faktyczną zgłoskę nie rozpoznawszy żadnego fonemu, co oznacza pominięcie lub zlanie zgłosek (linia przerywana na rys. 1),
- głowica przeszła do następnej faktycznej zgłoski rozpoznając zgłoskę (poprawnie lub błędnie – na rys. 1 linia ciągła).

Na ogół zakłada się, że prawdopodobieństwa poszczególnych przejść są jedynie funkcją rozpoznawanego fonemu i nie zależą ani od fonemów sąsiednich, ani od historii rozpoznawania. W przeciwnym razie zadanie rozpoznawania byłoby bardzo złożone algorytmicznie, a zadanie „treningu" systemu – uciążliwe ze względu na dużą wymaganą ilość materiału mówionego.

Zadaniem *treningu systemu* jest ustalenie prawdopodobieństw poszczególnych wspomnianych przejść, potrzebnych przy opracowaniu modeli fonetycznych poszczególnych słów (dokładniej mówiąc, poszczególnych grup słów zgodnych ze sobą fonetycznie). Trening systemu składa się z czterech etapów:
- wybranie reprezentatywnej próbki słów (tekstu) i dokonanie jego nagrania,
- oszacowanie na podstawie nagrania warunkowych prawdopodobieństw poszczególnych rodzajów błędów rozpoznawania, dla wszystkich kombinacji fonem rozpoznany – fonem rzeczywisty,
- opracowanie modelu Markowa dla całego tekstu (z uwzględnieniem dla każdego faktycznego fonemu tylko ograniczonej liczby, np. 5, najbardziej prawdopodobnych błędów rozpoznawania),
- optymalizacja macierzy prawdopodobieństw rozpoznawania fonemów w celu osiągnięcia najtrafniejszego rozpoznawania słów z tekstu treningowego.

W celu opracowania modelu każdego słowa należy uprzednio ustalić dla wszystkich kombinacji fonemów następujące prawdopodobieństwa:

- $P_{wstawienia}$ (*znak wstawiony/znak rzeczywisty*) – warunkowe prawdopodobieństwo wstawienia *znaku wstawionego* podczas pobytu głowicy na zgłosce *znaku rzeczywistego*, np. $P_{wstawienia}$ (*'i'/'e'*) to prawdopodobieństwo wstawienia fonemu *'i'* podczas rozpoznawania na zapisie fonetycznym fonemu *'e'*;

- $P_{pominięcia}$ (*znak rzeczywisty*) – bezwarunkowe prawdopodobieństwo „przeskoczenia" fragmentu sygnału odpowiadającego danemu *znakowi rzeczywistemu*, np. $P_{pominięcia}$ (*'h'*) to prawdopodobieństwo pominięcia fonemu *'h'* w trakcie rozpoznawania;

- $P_{podstawienia}$ (*znak podstawiony/znak rzeczywisty*) – warunkowe prawdopodobieństwo rozpoznania *znaku podstawionego* wówczas, gdy głowica faktycznie przesunęła się nad zapisem sygnału fonemu *znaku rzeczywistego*, np. $P_{podstawienia}$ (*'h'/'k'*) to prawdopodobieństwo rozpoznawania fonemu *'h'*, gdy faktycznie wystąpiło *'k'*.

Wartości te można uzyskać dokonując analizy frekwencyjnej przygotowanego tekstu mówionego. Należy jednak zwrócić uwagę na pewne ryzyko: zbyt rzadkie występowanie w próbce niektórych fonemów (szacowaniu podlega ok. 2500 parametrów) może prowadzić do błędnych wartości wyjściowych (zbyt duża liczba zer w macierzy rozpoznawania). Wartości te należy odpowiednio zwiększyć lub zmniejszyć tak, by dopuszczały one możliwe (a w próbce nie występujące) odmienne rozpoznania niektórych rzadkich fonemów.

Ocena statystyczna oddzielnych prawdopodobieństw rozpoznania poszczególnych fonemów jest pierwszym przybliżeniem pożądanych parametrów modelu Markowa. Celowe jest uwzględnienie „kontekstu" – chcemy rozpoznać nie fonemy, lecz całe słowa. Dlatego dokonuje się

następnie iteracji maksymalizujących dla danego tekstu treningowego prawdopodobieństwo warunkowe rozpoznania przez układ sekwencji fonemów pod warunkiem pojawienia się na wejściu tego układu sygnału mowy tekstu treningowego:

P (tekst rozpoznawany/tekst treningowy).

Dokonuje się tego stosując np. algorytm Bauma-Weicha, jak w pracy [1].

LITERATURA:

[1] Baum L. E.: Inequility and associated maximization technique in statistical estimation for probabilistic function of Markov-processes. Inequilitiem No. 3, pp. 1–8, 1972

[2] Erman L.D., Lesser V.R.: The HEARSAY II speech understanding system A tutorial. In W.A. Lea (Ed.): Trends in Speech Recognition. Prentice Hall Signal Processing Series, pp. 361–381, 1980

[3] Higgins A.L., Wohlford R.E.: Keyword recognition using template concatenation. ICASSP-85, pp. 1233–1236, 1985

[4] Kielczyński G.: Digital synthesis of speech and its prosodic features by means of a microphonemic method. Sprawozdania IIUW 65, Warszawa 1978

[5] Lea W.A.: Specific contributions of the ARPA SUR project. In W.A. Lea (Ed.): Trends in Speech Recognition. Prentice Hall Signal Processing Series, pp. 382–421, 1980

[6] Levenshtein V.I.: Binary/Codes capable of Correcting deletions, insertions and reversals. Cybernetics and Control Theory, No. 10, pp. 707–710, 1966

[7] Nieman H., Brietzmann A., Hein H. W., Muhlfeld R., Regel P., Schukat G.: A System for understanding continuons German Speech. Information Sciences, No. 33, pp. 87–113, 1984

[8] Nieman H., Brietzmann A., Muhlfeld R., Regel P., Schukat G.: The speech understanding and dialog system EVAR. In: DeMori R., Suen C.Y.: New Systems and Architectures for Automatic Speech Recognition and Synthesis NATO ASC Series F, pp. 271–302, Springer-Verlag, Berlin 1985

[9] Regel P.: Akustisch-phonetische Transformation fuer die automatische Spracherkennung. Dissertation am Lehrstuhl 5 (Mustererkennung), IMMD Universitaet Erlangen-Nuernberg, 1987

[10] Regel P.: Automatic extraction of phonetic symbols from the speech wave of fluently spoken German speech 6th. ICPR, pp. 966–968, 1982

[11] Schukat-Talamazzini E.G.: Generierung von Worthypothesen in kontinuierlicher Sprache. Informatik-Fachberichte 141, Springer-Verlag, Berlin-Heidelberg 1987

[12] Tadeusiewicz R.: Głosowe wprowadzanie informacji do komputera. Informatyka, nr 7, 8, 1987.

Docierają do redakcji sygnały z całego kraju o trudnościach z nabyciem INFORMATYKI w kioskach Ruchu.

Pragniemy wyjaśnić Czytelnikom przyczynę tych kłopotów. Redakcja podjęła decyzję zmniejszenia liczby egzemplarzy skierowanych do sprzedaży kioskowej, gdyż Ruch pobiera bardzo wysoką prowizję za swoją usługę, nie gwarantując prawidłowego rozdziału egzemplarzy w sieci kioskowej.

W związku z tym prosimy wszystkich zainteresowanych nabyciem INFORMATYKI o kontakt z redakcją (tel. 39-14-34) lub Działem Handlowym SIGMY (tel. 40-30-86) – będziemy Państwu wysyłać egzemplarze za zaliczeniem pocztowym – oczywiście w miarę naszych możliwości dość ograniczonych wysokością nakładu. Czytelnicy z terenu i okolic Warszawy mogą kupić poszczególne numery w Klubie SIGMY przy ul. Mazowieckiej 12, w księgarni „Domu Książki" przy ul. Mokotowskiej 51/53 oraz w lokalu redakcji.

Jednocześnie przypominamy, że najpewniejszą formą otrzymywania INFORMATYKI jest prenumerata, gdyż ona głównie kształtuje wysokość nakładu. Szczegóły jej zamawiania podajemy na stronie II okładki.

COIG

CENTRALNY OŚRODEK INFORMATYKI GÓRNICTWA
40–065 Katowice, ul. Mikołowska 100
tel. 574-777, teleks 00313711, telefaks 517-442

PROJEKTOWANIE I PROGRAMOWANIE SYSTEMÓW INFORMATYCZNYCH.
KOMPLETACJA, SPRZEDAŻ SYSTEMÓW I SPRZĘTU KOMPUTEROWEGO, KSEROKOPIAREK, TELEFAKSÓW, PODZESPOŁÓW I CZĘŚCI ZAMIENNYCH.
INSTALACJA, NAPRAWY I PRZEGLĄDY URZĄDZEŃ MIKROKOMPUTEROWYCH.
SZKOLENIE PROGRAMISTÓW, PROJEKTANTÓW I OPERATORÓW SPRZĘTU INFORMATYCZNEGO.

Oferujemy usługi w zakresie ujmowania i przetwarzania danych w trybie wsadowo-partiowym oraz zdalnym realizowane na komputerach: MERA-9150, ODRA-1300, ICL-1900, 2900, 39, RIAD.
Sprzęt mikrokomputerowy kompatybilny z IBM PC konfigurujemy zgodnie z potrzebami użytkownika.
Instalujemy sieci lokalne. Tworzymy systemy aplikacyjne.
Prowadzimy serwis sprzętu mikrokomputerowego w zakresie instalacji, napraw i przeglądów.
Dostarczamy wymagane oprogramowanie narzędziowe.
Szkolimy z zakresu obsługi mikrokomputerów, systemów operacyjnych, sieci lokalnych i teletransmisji, języków programowania oraz generatorów baz danych.
Projektujemy systemy informatyczne wg założeń odbiorcy lub na podstawie analizy jego potrzeb.
Dostarczamy oprogramowanie gotowych projektów z zastosowaniem dowolnego języka.

EO/943/89

TOMASZ ŻAKOWICZ
Instytut Informacji Naukowej, Technicznej i Ekonomicznej
Warszawa

CDS/ISIS narzędzie do zarządzania bazami danych (2)

W drugiej części artykułu omówiono użytkowanie bazy danych w trybie bezpośrednim, zarządzanie i rozszerzenia systemu CDS/ISIS.

UŻYTKOWANIE BAZY DANYCH W TRYBIE BEZPOŚREDNIM

Użytkowanie bazy danych w trybie bezpośrednim wymaga użycia programu firmy IBM o nazwie CICS (ang. *customer information control system*). Jest to system monitorujący wykonywanie wszystkich programów aplikacyjnych. Zainicjowanie pracy z bazą danych wymaga wprowadzenia jednego z poleceń

ISIS *NNNN PPPP*
ISIS *NNNN PPPP DBN*

gdzie *NNNN* oznacza identyfikator użytkowania, *PPPP* – hasło, *DBN* – nazwę bazy danych. Sesję kończy się poleceniem EXIT.

W środowisku ISIS są możliwe dwa tryby pracy: tryb wprowadzania danych oraz wyszukiwania. W trybie wprowadzania danych można korzystać z jednego z dwóch edytorów, mających również niemal niezależne środowiska programowe. W każdym z trybów jest dostępny dziedziczony z programu CICS mechanizm pełnego stronicowania. Oprócz poleceń przewidzianych dla trybów wprowadzania i wyszukiwania występuje pięć poleceń realizowanych bezkontekstowo:

CHANGE – zmiana bieżącej bazy danych na inną,
EXIT – zakończenie sesji,
INFO – wyświetlenie paneli informacyjnych,
HELP – udzielenie pomocy informacyjnej użytkownikowi,
SET – zmiana charakterystyki środowiska, tj. zakresów wyświetlania, języka komunikacji z użytkownikiem (angielski, francuski, niemiecki lub hiszpański), sposobu weryfikacji poprawności wprowadzanych indeksów oraz częstości aktualizacji pliku transakcyjnego.

Tryb wprowadzania danych

Tryb ten umożliwia wprowadzanie nowych rekordów oraz modyfikowanie i usuwanie istniejących. Wszelkie operacje są dokonywane w pliku transakcyjnym. Każda modyfikacja lub wprowadzenie nowego rekordu powoduje składowanie kopii tego rekordu w pliku roboczym. Po zakończeniu sesji wszystkie te rekordy są drukowane, tworząc kopie dokumentacyjne sesji dla poszczególnych użytkowników systemu. W trybie wprowadzania danych są drukowane następujące polecenia (nawiasy trójkątne pełnią funkcję metaograniczników):

<numer> – modyfikowanie rekordu o danym numerze;
P *<numer>* – tworzenie kopii dokumentacyjnej dla rekordu o danym numerze;
NEW – tworzenie nowego rekordu (automatyczne numerowanie);
WS *<name>* – wybranie arkusza wprowadzania (ang. *worksheet*) o danej nazwie;
WS=... – identyfikowanie arkusza wprowadzania;
REVISE – aktualizowanie rekordów wybranych przez zapytanie za pomocą edytora wierszowego EDIT;
RFULL – aktualizowanie rekordów wybranych przez zapytanie za pomocą edytora pełnoekranowego FULL;
DEFINE – definiowanie dynamiczne (w sesji) wartości domyślnych;
CLEAR – usuwanie dynamiczne wartości domyślnych;
BATCH – wprowadzanie modyfikacji do pliku indeksowego;
COPY – kopiowanie i redagowanie rekordu;
MOVE – kopiowanie z usunięciem wzorca i redagowanie rekordu;
THUP – aktualizowanie tezaurusa.

Możliwe jest zdefiniowanie arkuszy ułatwiających wprowadzanie danych, określających układ graficzny obrazu ekranu, a także podzbiór pól (w stosunku do zbioru wszystkich pól zdefiniowanych dla bazy danych w tablicy FDT), które należy wypełnić. Przy wprowadzaniu można sterować postacią wprowadzanych rekordów dokonując selekcji odpowiednich arkuszy. Dla danej bazy danych istnieje zawsze formularz standardowy. Upoważniony użytkownik może zdefiniować nowy formularz wprowadzania korzystając ze specjalnego, prostego języka definiowania zawartości ekranu.

Projektując bazę danych, można dla poszczególnych pól zdefiniować wartości domyślne, automatycznie wpisywane do każdego tworzonego rekordu, o ile wprowadzający nie wpisze do tych pól aktualnych danych. Użytkownik ma również możliwość dynamicznej zmiany poszczególnych wartości domyślnych na czas trwania swojej sesji, co ułatwia wprowadzanie rekordów należących do tej samej grupy tematycznej.

Polecenie BATCH umożliwia modyfikowanie zawartości słownika danej bazy. Słownik ten wchodzi w skład pliku indeksowego. Ponieważ wszelkie modyfikacje zawartości bazy danych są możliwe jedynie w trybie wsadowym, więc polecenie BATCH powoduje generowanie wsadowego zadania aktualizującego słownik, kierowanego do wykonania po zakończeniu sesji w trybie bezpośrednim. Każdy termin może być dodany do słownika, usunięty lub zastąpiony innym. Dla każdego terminu można również określić jego *status systemowy*. Termin może mieć następujący status:
- termin *kontrolowany* – musi być wprowadzony do słownika przed dołączeniem do niego odsyłaczy do rekordów, w których występuje ze statusem indeksów;
- termin *niekontrolowany* – jest automatycznie wprowadzany do słownika przez system;
- *synonim* – termin nie mający odsyłaczy do rekordów pliku głównego, lecz mający takie odniesienia do terminu preferowanego jako wartość indeksowa (deskryptor);
- *słowo stopujące* – nie ma wówczas odsyłacza do rekordów bazy danych; umożliwia jednak ochronę słownika przed automatycznym wprowadzaniem terminów mających znikomą wartość dyskryminacyjną dla wyszukiwania.

Istnieje też możliwość zmiany statusu oraz definiowania przez użytkownika dodatkowych statusów terminu.

Polecenia MOVE i COPY umożliwiają tworzenie nowego rekordu na podstawie rekordu istniejącego. Kopiowanie rekordu można realizować z tej samej lub z innej bazy danych (pod warunkiem, że plik główny rezyduje w tym samym klastrze). Kopiowanie znalezionego rekordu można również zrealizować korzystając z funkcji kopiowania „C/" programu CICS. Jako miejsce przeznaczenia można wybrać dowolny terminal aktywny w sesji. Do modyfikowania zawartości wyszukanego rekordu służą następujące polecenia redakcyjne:
CHA – zmiana zwartości pola;
ADD – dodanie nowego pola;
ADR – dodanie nowego wystąpienia pola;
DEL – usunięcie pola;
REP – zamiana pola;
TAG – zmiana etykiety pola;
MOD – zmodyfikowanie zawartości pola (dla terminali typu 3270);
TRA – certyfikacja rekordu;
DEL ** – usunięcie rekordu;
D – wyświetlenie rekordu;

FULL – zmodyfikowanie zawartości rekordu (dla terminali typu 3270);
CANCEL (lub C), END (lub E) – zakończenie przetwarzania rekordu. Dla terminali typu IBM 3270 lub zgodnych z nimi istnieje możliwość edycji pełnoekranowej – nie tak swobodnej jak we współczesnych edytorach mikrokomputerowych, ale znacznie upraszczającej pracę.

Tryb wyszukiwania

W trybie wyszukiwania użytkownik może wybrać interesującą go bazę danych i grupę rekordów spełniających zadane kryteria. Dysponuje on następującymi poleceniami (w nawiasach podano minimalne skróty):
= <*zapytanie*> – podanie liczby dokumentów „spełniających" zapytanie;
DISPLAY (D) – wyświetlanie wyszukanych rekordów;
EXECUTE (EX) – wyszukanie według istniejącego zapytania;
EXPLORE (EXP) – wyświetlenie fragmentu pliku indeksowego;
FLUSH (F) – usunięcie z systemu wszystkich formuł wyszukiwawczych (wraz ze składowymi);
KEEP (KEEP) – zapamiętanie zapytania lub wyświetlenie listy pamiętanych zapytań;
LIMITS (L) – zdefiniowanie zakresu wyszukiwania;
RECALL (R) – wyświetlenie aktualnych formuł wyszukiwawczych;
SAVE (S) – zapamiętanie bieżącej formuły dla wyszukiwania wsadowego;
SORT – sortowanie wyniku wyszukiwania;
SUBMIT – wyświetlenie zbioru zapytań zdefiniowanych przez danego użytkownika;
TEXT (T) – wyszukanie według rozszerzonych formuł wyszukiwawczych;
THES – przeszukanie tezaurusa.
Polecenia te mogą być wprowadzane w każdej chwili trwania sesji.

Polecenie DISPLAY umożliwia wyświetlenie wyniku wyszukiwania w uprzednio lub aktualnie zdefiniowanym formacie. Służy ono również do wyświetlania dowolnych rekordów bazy danych. Na podstawie jednoznacznie przypisanych poszczególnym zapytaniom numerów, polecenie EXECUTE powoduje wyszukanie i wykonanie formuły wyszukiwawczej bez konieczności powtórnego jej kodowania. Umożliwia też wyszukiwanie według tej samej formuły w różnych bazach systemu (do ich zmiany używa się polecenia CHANGE) lub w różnych obszarach bazy danych (zmianę zakresu przeszukiwania uzyskuje się poleceniem LIMITS). Polecenie SAVE kieruje zapytanie do kolejki tych formuł wyszukiwawczych, które będą przetwarzane w trybie wsadowym po zakończeniu sesji w trybie bezpośrednim. Ze względu na mniejsze wykorzystanie zasobów systemowych koszt wyszukiwania w trybie wsadowym jest znacznie niższy.

Język wyszukiwania

Podstawą tego języka jest algebra boolowska. Każdy deskryptor lub wartość indeksowa zapisana w jednym z rekordów bazy danych jest traktowana jako reprezentant zbioru tych rekordów, w których występuje. W wyrażeniach boolowskich mogą występować operatory alternatywy, koniunkcji i negacji odpowiednio *, +, ¬

W formułowaniu wyrażenia boolowskiego użytkownik może używać trzech rodzajów identyfikatorów grupy rekordów bazy danych:
- wartości indeksowych,
- wartości indeksowych uciętych z prawej strony,
- *terminów ANY*.

Wartością indeksową może być każdy fragment tekstu wybieralny jedną z dostępnych w systemie ISIS technik ekstrakcji wartości indeksowych, np. deskryptor, słowo kluczowe, fraza kluczowa, nazwisko autora, kod. Lista wartości indeksowych powinna być znana przed rozpoczęciem wyszukiwania. Specyfikowanie identyfikatora pola, z którego powinna być ekstrahowana wartość indeksowa odbywa się według formatu:

<*deskryptor*>/(*t*1, *t*2, *t*3,...)

gdzie *t*1, *t*2, *t*3 są identyfikatorami pól branych pod uwagę dla wartości indeksowej określonej przez <*deskryptor*>.

Zamiast specyfikacji pojedynczej wartości indeksowej można wyspecyfikować korzeń leksykograficzny. Technika ta, zwana *prawostronnym ucięciem*, pozwala wybierać do formuły wyszukiwawczej wartości indeksowe według początkowej sekwencji liter ich zapisu. Poszczególne wartości indeksowe z grupy tych wszystkich, które mają identyczną początkową sekwencję liter są łączone w wyrażeniu wyszukiwawczym operatorem alternatywy. Wynik wyszukiwania stanowią te rekordy, które są identyfikowane przez co najmniej jeden termin grupy. Specyfikacja korzenia polega na umieszczeniu znaku $ bezpośrednio za kluczową sekwencją. Przykładowo, jeżeli w słowniku występują wartości indeksowe:

FILM

FILM INDUSTRY
FILM LIBRARIES

to specyfikacja:

FILM$

będzie równoważna formule:

(FILM*FILM INDUSTRY*FILM LIBRARIES).

Termin ANY jest terminem generatywnym reprezentującym uprzednio zdefiniowaną grupę wartości indeksowych. Użycie terminu ANY powoduje sumowanie teoriomnogościowe wszystkich zbiorów rekordów odpowiadających wartościom indeksowym wchodzącym w skład danej grupy ANY, podobnie jak w przypadku wartości indeksowych prawostronnie uciętych. Nie wszystkie wartości indeksowe formuły wyszukiwawczej muszą mieć identyczny status informacyjny (priorytet). Za pomocą techniki *kropkowania* jest możliwe przypisanie niektórym składnikom formuły wyszukiwawczej wyższego priorytetu. Przykładowo, formuła:

.EDUKACJA KOBIET + WŁOCHY

spowoduje wyszukanie tylko tych rekordów, w których edukacja kobiet jest wartością indeksową szczególnie istotną. Kropką można również poprzedzać grupy ANY. Na przykład:

.ANY KRAJE BENELUXU

jest równoważne

.BELGIA*.HOLANDIA*.LUXEMBURG

Technika kropkowania wymaga użycia odpowiedniej techniki ekstrakcji wartości indeksowych.

Operator alternatywy powoduje sumowanie teoriomnogościowe grup rekordów odpowiadających poszczególnym wartościom indeksowym wchodzącym w skład alternatywy. Operator koniunkcji reprezentuje iloczyn teoriomnogościowy, a zatem realizuje zawężenie zbioru odpowiedzi w stosunku do każdego ze zbiorów odpowiadających poszczególnym wartościom indeksowym występującym w koniunkcji. W wyrażeniu $A \neg B$ operator negacji powoduje selekcję tych rekordów, które zawierają termin *A*, lecz nie zawierają terminu *B*.

Następujące operatory dodatkowo precyzują zbiory odpowiedzi:
(F) – dla danego pola musi istnieć takie pojedyncze wystąpienie pola, które zawiera obydwa terminy (tj. termin poprzedzający operator i termin następujący po nim);
(G) – dla danego pola muszą istnieć jego wystąpienia zawierające obydwa terminy, przy czym nie muszą być one zawarte w tym samym wystąpieniu pola;
... – terminy muszą być oddalone od siebie we wspólnej frazie o nie więcej niż tyle słów, ile występuje kropek minus jeden:
$$$ – podobnie, jak w przypadku kropek, z tym że w tym wypadku między wartościami indeksowymi połączonymi operatorem musi być dokładnie tyle słów, ile jest znaków dolara minus jeden. Wyrażenia boolowskie są tworzone z wartości indeksowych i operatorów. Elementy wyrażeń mogą być ujęte w nawiasy.

Wyrażenie boolowskie poprzedzone znakiem równości stanowi formułę wyszukiwawczą (zapytanie), np.

=SZTUKA+(WŁOCHY*FRANCJA)

W odpowiedzi system przypisze zapytaniu jednoznaczny numer, po

czym wyświetli informację o liczbie odsyłaczy do tych dokumentów, dla których formuła wyszukiwawcza jest spełniona.

Odmienną technikę wyszukiwania zapewnia polecenie TEXT. Wyszukiwanie odbywa się nie na podstawie pliku indeksowego, lecz bezpośrednio na zawartości pliku głównego. Oczywiście efektywne przetwarzanie formuł poleceniem TEXT wymaga wyspecyfikowania niewielkich zakresów przeszukiwania i (lub) dysponowania bardzo szybkim komputerem. Formuły wyszukiwawcze w tym przypadku mogą być rozszerzone. Dopuszczalne jest specyfikowanie dowolnych sekwencji znakowych, a nie tylko aktualnych wartości indeksowych, można używać operatorów relacyjnych: $<$, $>$, $=$, $\neg$, w odniesieniu do sekwencji znaków dowolnego pola traktowanych jako liczba i dowolnych stałych arytmetycznych naturalnych. Można wyszukiwać rekordy w zależności od obecności lub nieobecności w nich poszczególnych pól. Oczywiście wyszukiwanie według klasycznych formuł logicznych jest w formułach przetwarzanych poleceniem TEXT również dozwolone.

ZARZĄDZANIE W SYSTEMIE BAZY DANYCH

Operacje podejmowane przez administratora systemu baz danych można podzielić na jednorazowe, takie jak zainicjowanie bazy danych lub odtwarzania zawartości bazy w razie awarii w systemie, oraz periodyczne, np. aktualizowanie plików (głównego i indeksowego), tworzenie kopii archiwalnych, kontrola upoważnień dostępu, wymiana zawartości bazy z innymi systemami.

Inicjowanie bazy danych

Do zainicjowania plików systemowych CDS/ISIS służą makrodefinicje ALLOCATE i INITF. Wymagają one podania kilku specyfikacji systemowych, takich jak: typy urządzeń wchodzących w skład konfiguracji komputera, prefiksy identyfikacyjne nazw tworzonych plików, szacunkowe wskaźniki średniego rozmiaru rekordu w pliku głównym i liczebności bazy. W wyniku wywołania tych makrodefinicji otrzymuje się sekwencję kart w języku JCL inicjujących pliki systemowe na nośnikach magnetycznych. Jeśli tworzona baza jest pierwszą bazą w systemie, to należy również zainicjować środowisko konwersacyjne systemu. Wymaga to co najmniej zdefiniowania tablic PPT, PCT, FCT i SNT dla programu CICS (w starszych wersjach systemu CDS/ISIS dla systemów komputerowych, opartych na jednostkach Riad, CICS może być zastąpiony systemem KAMA).

Zbiór procedur katalogowanych stanowiący element systemu CDS/ISIS zakłada, że wszystkie taśmy kopii rezerwowej (ang. *backup*) są zorganizowane jako system GDG (ang. *generation data group*). W systemie tym najnowsza kopia rezerwowa danego pliku jest zapisywana w miejsce najstarszej. GDG inicjuje się przez utworzenie katalogu oraz uruchomienie zadania wygenerowanego przez trzecią, opcjonalną z punktu widzenia możliwości prawidłowego funkcjonowania systemu, makrodefinicję BKUP.

Aktualizacja plików głównych bazy danych

Uaktualnienie plików głównych bazy danych odbywa się w trybie wsadowym. Początkowo jest aktualizowany plik główny (MASTER), a w razie powodzenia tej operacji – również plik indeksowy (INVERTED). Podczas aktualizowania pliku indeksowego jest systemowo blokowana możliwość zapisu do pliku głównego, co nie dopuszcza do powstania niezgodności między zawartością tych dwóch plików. Zamknięcie bazy dla zapisu występuje również w przypadku awarii procesu aktualizacji. Ponowne odblokowanie bazy wymaga na ogół powtórzenia poszczególnych jego etapów. Jeśli w systemie nastąpiła utrata części lub całości danych pliku głównego lub indeksowego, to po zablokowaniu bazy następuje wczytanie ostatniej kopii archiwalnej i reorganizacja bazy. Następnie baza jest aktualizowana na podstawie zapisu w dzienniku pliku głównego i indeksowego. Taka strategia odzyskiwania danych po awarii pozwala na odtworzenie prawie 100% informacji. Aby baza danych zawierała aktualne informacje administrator systemu musi dostatecznie często przeprowadzać operacje aktualizacyjne. Ponieważ w obecnej wersji systemu użytkownicy mają możność bezpośredniego aktualizowania bazy w trybie wsadowym, od administratora wymaga się przede wszystkim aktualizowania bazy na podstawie zawartości pliku transakcyjnego, w których są składowane polecenia modyfikacyjne z przetwarzania w trybie konwersacyjnym. Do obowiązków administratora należy też okresowe tworzenie kopii archiwalnych. Jeśli sporządzenie takiej kopii dokonało się bez zakłóceń, to zawartość dziennika jest automatycznie zamazywana.

Wymiana zawartości bazy danych

CDS/ISIS umożliwia konwersję dwustronną zawartości bazy z (i do) formatu wymiennego ISO 2709. Dzięki temu można dość tanio „włączać" bazy danych zarządzane innym SZBD do systemu baz danych CDS/ISIS lub odwrotnie. Przykładem zastosowania konwersji może być system sieciowy, którego centrum stanowi duża baza danych, a funkcję terminali pełnią mikrokomputery IBM PC/XT (lub AT) z systemem CDS/ISIS w wersji „mikro" umożliwiającym efektywne przetwarzanie baz o rozmiarze do kilku tysięcy rekordów. Terminale te mogą pełnić funkcję stacji wprowadzania danych lub też samodzielnie przetwarzać dowolne podbazy głównej bazy systemu.

Kontrola upoważnień dostępu do bazy danych

W systemie CDS/ISIS zastosowano wielostopniowy system ochrony. Pierwszy stopień stanowi ochronę hasłową przed bezpośrednim dostępem do środowiska CDS/ISIS. Hasło jest definiowane w rekordzie UID składowanym w bibliotece tablic FDT. Następny stopień ochrony realizują karty autoryzacyjne USR, będące elementem definicji bazy danych. Użytkownik może wykonywać w systemie operacje zgodne z jego kartą autoryzacji w zależności od wartości kodu statusu dostępu. Jeśli kodem jest znak *S*, to jest dopuszczalne wyszukiwanie i przeglądanie zawartości bazy danych, *M* umożliwia modyfikowanie zawartości rekordów, *D* – usuwanie lub modyfikowanie zawartości rekordów, *C* – usuwanie lub modyfikowanie zawartości rekordów, a także tworzenie nowych rekordów. Kody *M*, *C*, *D* mogą być łączone z *S*. Jeśli karty autoryzacyjne nie są w systemie implementowane, to dodatkową ochroną przed zmianą zawartości bazy jest zdefiniowane w tablicy FDT hasło blokujące wprowadzanie danych do systemu. Ochrona poszczególnych pól rekordów bazy przed nieupoważnionym odczytem polega na nieinformowaniu użytkownika o nazwach pól chronionych i hasłowej ochronie formatów wydruku (wcześniej zdefiniowanych), które nie powinny być przez niego stosowane. Dodatkowo hasłem jest chroniony podsystem tezaurusowy. Wszystko to wymaga od użytkownika z wysokim priorytetem dostępu niezłej pamięci do haseł lub noszenia ze sobą stosownej ściągawki. Zadanie dla administratora bazy danych polega na prowadzeniu ewidencji haseł, a więc na ich przydzielaniu, unieważnianiu oraz zmianie statusów dostępu.

ROZSZERZENIA PAKIETU CDS/ISIS

Oprócz podsystemów realizujących standardowe funkcje SZBD w pakiecie znajdują się podsystemy rozszerzające użyteczność oprogramowania:

- DML (ang. *data manipulation language*) – język manipulowania zawartością bazy danych ze środowiska PL/I;
- podsystem fotokompozycji – w ośrodkach wyposażenia w urządzenie do fotoskładu DIGISET 42T2 umożliwia przenoszenie wydruków wyszukanych dokumentów na matryce offsetowe w atrakcyjnej szacie graficznej;
- podsystem obsługi tezaurusa – umożliwia utrzymywanie tezaurusa w formie maszynowej, a także czyni go dostępnym przy pracy z bazą danych w trybie bezpośrednim.

Język DML

Współczesne systemy tworzenia i obsługi baz danych umożliwiają dostęp do struktur bazy ze środowiskiem języka wysokiego poziomu. Dzięki temu systemy te stają się bardziej elastyczne, rozszerza się zakres ich zastosowań, a także upraszcza się proces adaptacji oprogramowania do specyficznych wymagań użytkowników. Ponieważ CDS/ISIS początkowo takiego dostępu nie miał, w INTE opracowano system biernego dostępu do struktur danych bazy z poziomu języka PL/I. System ten umożliwia odczytywanie poszczególnych rekordów bazy. Informacje zawarte w tych rekordach mogą być następnie przetwarzane przez dowolne programy użytkowe napisane w PL/I. W wyniku połączenia systemu biernego dostępu do struktur bazy i języka PL/I został utworzony język manipulowania danymi bazy, nazwany przez autorów DML.

System biernego dostępu stanowią następujące procedury:
ISISOPN – otwarcie konkretnej bazy danych systemu,
ISISCLS – zamknięcie bazy i zwolnienie zasobów,
ISISANS – wybranie rekordów z bazy na podstawie wyrażenia języka wyszukiwania CDS/ISIS,
ISISFLD – zrealizowanie dostępu do pól rekordów wyszukanych przez

ISISANS,
ISISFLP – zidentyfikowanie parametrów rekordu,
ISISPRT – drukowanie rekordów z pliku utworzonego przez ISISANS, według dowolnego predefiniowanego formatu.

Procedury te umożliwiają dość łatwe tworzenie nawet złożonych aplikacji, np. rozszerzenie wyszukiwania opartego na wyrażeniach boolowskich do wyszukiwania logiczno-relacyjnego (stosowanego w relacyjnych bazach danych). Oczywiście, moc systemu dostępu wzrosłaby znacznie po zaimplementowaniu procedur umożliwiających czynną manipulację zawartością bazy z poziomu języka PL/I, tj. dopuszczających zmianę zawartości poszczególnych rekordów.

Podsystem fotokompozycji

Podsystem fotokompozycji umożliwia bardzo precyzyjne tworzenie szaty graficznej dokumentów przeznaczonych do druku. Dopuszcza on wykonywanie wielu operacji, m.in.: zmianę krojów znaków graficznych, środkowanie fragmentów tektu, definiowanie formatu strony (jak np. definiowanie marginesów poziomych i pionowych oraz nagłówków). Możliwości podsystemu są w pełni wykorzystywane przez współpracujące z nim urządzenie do fotoskładu DIGISET 40T2. Szczęśliwie dla ośrodków nie wyposażonych w to urządzenie część funkcji (niezależnych sprzętowo) można zrealizować z użyciem zwykłych drukarek wierszowych.

Podsystem obsługi tezaurusa

Przedstawione powyżej dwa rozszerzenia CDS/ISIS, język DML i podsystem fotokompozycji funkcjonowały już w niektórych poprzednich wersjach systemu. Nowością jest natomiast podsystem obsługi tezaurusa.

Eksploatacja bibliograficznej bazy danych staje się znacznie dogodniejsza i tańsza, jeśli w systemie występuje tezaurus – rodzaj słownika zawierający zarówno listę deskryptorów, jak i łączące je ze sobą relacje. Użycie tezaurusa umożliwia tworzenie klasyfikacji tematycznych dla poszczególnych pozycji bibliograficznych. Fizycznie tezaurus stanowi nieco zmodyfikowaną ISIS-ową bazę danych. *Słownikiem* tezaurusa jest nazywany zbiór tych terminów, które mają rekord w pliku głównym – są to tzw. *terminy bazowe*. Zbiór terminów bazowych jest na ogół mniej liczny od zbioru wszystkich terminów występujących w tezaurusie.

Podsystem obsługi tezaurusa realizuje następujące funkcje:
– tworzenie i utrzymywanie tezaurusa,
– przeglądanie tezaurusa w trybie bezpośrednim, podczas korzystania z dowolnej bazy danych z nim związanej,
– formułowanie zapytań na podstawie tezaurusa, wykonywanych po wyjściu z podsystemu.
Do realizacji tych funkcji są przeznaczone dwa polecenia systemowe *THES* i *THUP*. Użycie polecenia *THUP* zawierającego funkcjonalnie *THES* jest chronione hasłem. Polecenie *THES* umożliwia:
- stronicowanie zawartości bazy przy wyświetlaniu według:
– deskryptora,
– terminu będącego w wyspecyfikowanej relacji z danym deskryptorem,
– skróconego deskryptora,
– odsyłacza deskryptorowego, zawierającego numer rekordu, w którym jest zapisany dany deskryptor,
– dowolnego deskryptora, występującego na uprzednio wyznaczonej liście deskryptorów związanych z danym wyspecyfikowaną relacją;
- konstruowanie zapytania na podstawie zawartych w tezaurusie deskryptorów;
- dodawanie do poszczególnych terminów prefiksów eliminujących trudności wynikające z obecności homonimów w tworzeniu precyzyjnych zapytań.

Polecenie THUP umożliwia dodatkowo:
- wprowadzanie do tezaurusa nowych deskryptorów lub relacji;
- usuwanie deskryptorów lub relacji;
- zmianę nazw deskryptorów;
- modyfikowanie deskryptorów typu *komentarz*.
Zbiór relacji deskryptorowych obsługiwanych przez system składa się z podzbioru relacji systemowych i podzbioru relacji definiowanych przez użytkownika (w szczególności pustego).

Każdemu terminowi wprowadzonemu do słownika tezaurusa jest automatycznie nadawany status *terminu bazowego* (ang. *top term*). Z terminem takim może być związany: relacją *BT* (ang. *broader term*) – inny termin o szerszym znaczeniu, relacją *NT* (ang. *narrower term*) – termin o węższym znaczeniu, relacją *RT* (ang. *related term*) termin pokrewny (nie związany z relacją hierarchiczną) lub relacją *US* – termin synonimiczny, lecz wycofany z użycia. Z terminem bazowym może być też związany dowolny fragment tekstu nazywany *deskryptorem komentarzowym*.

Repertuar relacji definiowalnych przez użytkownika jest dosyć ograniczony – mogą to być jedynie relacje komentarzowe, w których termin bazowy może być „wiązany" jedynie z deskryptorem komentarzowym, oraz relacje asymetryczne, w których termin związany z bazowym nie występuje w słowniku tezaurusa, aczkolwiek posiada referencję do terminu bazowego.

• • •

Pakiet CDS/ISIS został zainstalowany, przeważnie w starszych wersjach, w wielu krajowych ośrodkach przetwarzania danych. Radosne zdumienie budzi fakt jego bezpłatnej dystrybucji. Bezpłatne były również kursy szkoleniowe i usługi serwisowe świadczone przez INTE. Oprogramowanie jest dostarczane użytkownikowi wraz z wersją źródłową umożliwiającą wprowadzanie modyfikacji, a także wraz z pełną dokumentacją. W tej sytuacji trudno wyobrazić sobie bardziej korzystne warunki dla potencjalnych użytkowników bibliograficznych baz danych.

Bezpieczeństwo danych

dokończenie z III strony okładki

Oprócz przepisów zawartych w ustawach o spółkach oraz działalności gospodarczej, tajemnicę firmy w RFN chronią w pierwszym rzędzie trzy paragrafy (§ 1, 17 i 18) ustawy o zwalczaniu nieuczciwej konkurencji (*UWG – Gesetz gegen den unlauteren Wettbewerb*). Szczególnie zaskakują postanowienia § 17, ponieważ dotyczą nie tylko ochrony przed firmami konkurencyjnymi, ale przed własnymi pracownikami. Przepis ten zabrania przekazywania na zewnątrz tajemnic firmy przez cały okres trwania zatrudnienia. Rozszerzenie tego okresu jest możliwe jedynie przez wprowadzenie odpowiedniej klauzuli do umowy o pracę.

Pracownikom, którzy naruszą postanowienia § 17 grozi kara pieniężna oraz pozbawienia wolności do lat 5, niezależnie od konieczności zapłacenia odszkodowania za wyrządzone szkody. Podobną, choć nieco mniejszą odpowiedzialność przewiduje § 18 (kara pieniężna, pozbawienie wolności do lat 2, zapłacenie odszkodowania). Dotyczy on kontrahentów przedsiębiorstwa, którzy powierzone im dokumenty techniczne wykorzystują w sposób niezgodny z otrzymanym zleceniem lub przekazują je w inne ręce. Paragraf ten jest rzadko stosowany i zawiera istotną lukę, ponieważ dotyczy dokumentów o charakterze technicznym, a całkowicie pomija dokumenty ekonomiczne, które w większości przypadków są przedmiotem kradzieży. Sprawę dodatkowo komplikuje fakt, że w informatyce wyjątkowo często występują sytuacje, w których specjaliści z zewnątrz zatrudnieni przy uruchamianiu systemów, mają praktycznie nieskrępowany dostęp do wszystkich zasobów informacyjnych przedsiębiorstwa, w tym również do informacji najbardziej poufnych.

Jeżeli §18 jest niedostatecznie skuteczny w ściganiu sprawców kradzieży, to istnieje możliwość zastosowania §1. Przepis ten ma zapobiegać, aby tajemnice uzyskane w sposób legalny, nie zostały wykorzystane w sposób godzący w interesy firmy. Jego ostrze jest skierowane głównie przeciw tym byłym pracownikom, którzy w sposób godzący w dobre obyczaje sprzedają pozyskaną technologię lub na jej podstawie organizują firmę konkurencyjną. Ponieważ w tym przypadku udowodnienie przestępstwa jest wyjątkowo trudne, to również sankcje są znacznie łagodniejsze i ograniczają się jedynie do obowiązku wynagrodzenia szkód.

Komentując powyższe przepisy trudno pozbyć się refleksji, że najskuteczniejszym sposobem zabezpieczenia danych przed kradzieżą jest takie związanie pracownika z interesem firmy, aby odsunąć od niego wszelkie pokusy, jakie może zaoferować konkurencja. **(WK)**

JACEK MALEC
Instytut Cybernetyki Technicznej
Politechnika Wrocławska

Maszyny lispowe (2)

Na przełomie lat 1983–1984 kilka firm wprowadziło na rynek osobiste maszyny lispowe ML. Najbardziej popularne z nich to Xerox Dolphin i Dorado (z językiem Inter Lisp-D i systemem Loops), LMI Lambda, Symbolics 3670 (z językami Zeta Lisp i Common Lisp i systemem Zmacs, a także Explorer firmy Texas Instruments.

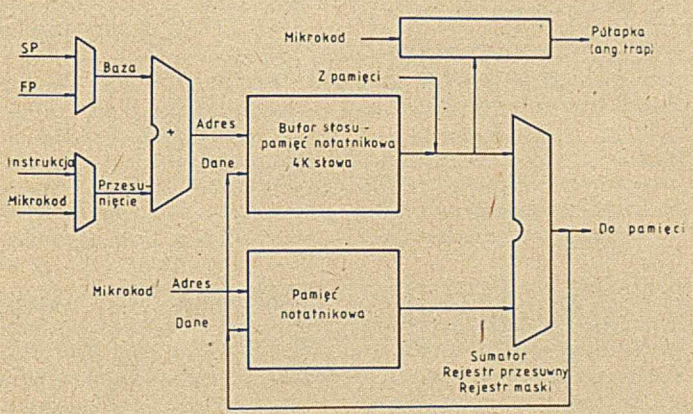
Z punktu widzenia użytkownika możliwości obliczeniowe tych komputerów są podobne. Zapewniają one dobry system operacyjny w Lispie, bardzo wygodny edytor, kompilator i (już nie wszystkie) mikrokompilator Lispu, mysz i okienka umożliwiające prosty dostęp do wszystkich udogodnień systemu, dużą pamięć wirtualną oraz sprzęg sieci Ethernet. Ceny tych komputerów kształtują się w okolicach 20–150 tys. dolarów USA.

Poniżej krótko omówiono architektury czterech spośród wspomnianych komputerów.

SYMBOLICS 3600

Komputery serii 3600 (np. 3620, 3670) firmy Symbolics [8, 21, 27] są, podobnie jak CADR, maszynami mikroprogramowanymi, ale przeznaczonymi już od chwili powstania założeń ich architektury do przetwarzania struktur listowych wyłącznie Lispu. Stąd nieco inne rozwiązania poszczególnych elementów jednostki centralnej, dużo więcej udogodnień sprzętowych oraz nieco inny mechanizm sterowania.

Podstawową strukturą sterowania jest wyłapywanie (ang. *trapping*), oznaczające sprzętową reakcję na pojawianie się określonych wzorców w słowie danych lub słowie instrukcji. Sterowanie i przepływ danych (rys. 1) odbywają się równolegle, co pozwala na przyspieszenie obliczeń w porównaniu z CADR, choć podstawowy cykl procesora jest dłuższy i wynosi od 180 do 250 ns.

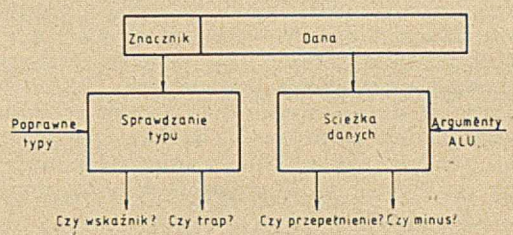


Rys. 1. Ścieżka danych w komputerach serii 3600

Pamięć mikrokodu zajmuje 16 K słów 100-bitowych – mikroinstrukcje mają charakter w pełni równoległy. Pamięć programów zajmuje od 256 K słów do 7,5 M słów; adres wirtualny zajmuje natomiast 28 bitów, co pozwala zaadresować 256 M słów, z których 5 M zajmuje oprogramowanie systemowe. Słowo w komputerach serii 3600 ma długość 36 bitów i może być jednego z formatów: 32 bity bezpośredniej danej (liczba), 2 bity znacznika i 2 bity kodu CDR (por. opis CADR) lub 28 bitów adresu, 6 bitów znacznika i 2 bity kodu CDR. Formaty te są uzupełnione o 7 bitów kodu korekcyjnego i jeden bit nieużywany, tak więc wynikowa szerokości ścieżki danych wynosi 44 bity.

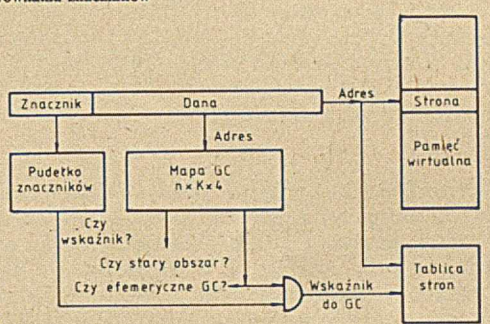
W maszynach Symbolics wprowadzono arytmetykę 32-bitową. Sprzętowe wspomaganie przetwarzania struktur Lispu obejmuje:

- definicje formatów danych,
- wyłapywanie typu danych[1] (ze znacznika, rys. 2),
- porównywanie typów dwóch danych (rys. 3),
- sprzętowo wspomagane operacje na stosach[2],



Rys. 2. Wyłapywanie typu danych

Rys. 3. Porównania znaczników



Rys. 4. Sprzęt wspomagający GC

- utworzenie kolejki odwołań do pamięci,
- wyłapywanie obszarów (rys. 4) aktualnie podlegających GC.

W 3600 zastosowano udoskonaloną wersję odzyskiwania nieużytków (GC) działającego w czasie rzeczywistym, nazwaną *efemerycznym odzyskiwaniem nieużytków*.

LMI LAMBDA

ML Lambda firmy LMI [8, 25] jest podobnie jak CADR przykładem maszyny o architekturze uniwersalnej, nie dostosowywanej sprzętowo,

[1] Sprawdzanie typu odbywa się podczas wykonywania instrukcji, a nie przed, jak w maszynach CADR. Na przykład makrooperacja + jest tylko jedna i deklaracje typów liczb w kompilatorze są pomijane. Lispowa funkcja ADD sprawdza typ argumentu w trakcie wykonywania i w przypadku liczb stałoprzecinkowych kończy działanie, dla liczb zmiennoprzecinkowych następuje – dzięki mechanizmowi wyłapywania – wykonanie odpowiedniego programu w mikrokodzie, a dla dużych liczb – programu w Lispie.

[2] 3600 nie ma rejestrów ogólnego przeznaczenia dostępnych na poziomie makrokodu i większość instrukcji używa stosów do pobierania argumentów i składania wartości. Ponadto każdy z aktywnych procesów ma powiązane ze sobą oddzielne środowisko, nazywane *grupą stosową*. Stąd też uwaga zwrócona na operacje stosowe.

a tylko mikroprogramowo do wymagań Lispu. Lambda zawiera 32-bitowy procesor z pamięcią mikroprogramu wielkości 64 K słowa 64-bitowe. Czas cyklu wynosi 200 ns. Lambda może zaadresować 32 M słowa pamięci, ma dużą liczbę rejestrów wewnętrznych, możliwość operowania bajtami, 4096-słową pamięć notatnikową instrukcji i danych, 2048-słowową pamięć notatnikową stosu, logikę porównywania znaczników oraz kolejkowanie 16-bitowych kodów rozkazów tworzonych przez kompilator.

Cechy wyróżniające maszynę Lambda to: możliwość mikrokompilacji oraz stronicowana pamięć mikrokodu. Mikrokompilacja pozwala uniknąć straty czasu poświęcanego na obsługę makrokodu oraz znacznie przyspiesza przekazywanie sterowania i (na ogół) przetwarzanie danych. Jak już wcześniej wspominano, szybka obsługa wywołania funkcji jest podstawą efektywnego obliczania programów Lispowych. W ML Lambda mikrokompilator może generować wyjątkowo szybkie wywołania funkcji, nazywane wywołaniami mikro-mikro. Wywołania i powroty mikro-mikro zajmują tylko po jednej mikroinstrukcji i są obsługiwane w ciągu 200 ns.

Aby utworzyć optymalny kod, mikrokompilator potrzebuje specjalnych deklaracji, analogicznych do wymagań typowych kompilatorów Lispu na zwykłych komputerach. Wywołania mikro-mikro są tworzone pod warunkiem podania odpowiednich deklaracji. Po pierwsze, mikrokompilator potrzebuje zapewnienia, że wywoływana funkcja w rzeczywistości znajdzie się w pamięci mikrokodu w chwili wywołania. Po drugie, wywołanie mikro-mikro korzysta ze stosu adresu mikrokodu o pojemności 256 słów. Funkcje z głęboką rekurencją muszą testować przepełnienie tego stosu – jest to powodowane przez odpowiednią deklarację.

Niektóre konstrukcje języka nie mogą pojawić się w funkcjach mikrokompilowanych. Dotyczy to argumentów typu &REST, konstrukcji sterujących CATCH i THROW oraz zwracania i pobierania wielu wartości. Ograniczeniem jest też fakt, że w pamięci mikrokodu może być tylko 64 K mikroinstrukcji, chociaż znacznie więcej może rezydować w pamięci głównej i być ładowane na jawne żądanie. Jednak 64 K jest z reguły wystarczające – oprogramowanie systemowe Lambda zajmuje 14 KB pamięci mikrokodu.

Procesor Lambdy komunikuje się z otoczeniem przez magistralę NuBus[3] z zegarem 10 MHz, o przepustowości 37,5 M bajtów/s. NuBus pozwala także na dołączenie wielu procesorów (ogólniej – pakietów aktywnych) o równym prawie dostępu do magistrali, co pozwoliło LMI oferować konfiguracje ML z wieloma procesorami Lambda dzielącymi wspólne zasoby [25].

Duża część efektywności Lambdy wynika z zastosowania pamięci notatnikowej dla stosu. Mikrokod dopilnowuje, aby znajdował się w niej szczyt stosu sterowania bieżącego zadania. Ramka wywołania aktualnej funkcji zawsze znajduje się w tej pamięci i dostęp do jej elementów (argumenty, zmienne lokalne, wartości pośrednie, nagłówek) odbywa się przez indeksowanie wskaźnika (umieszczonego w specjalnym rejestrze) do ramki.

XEROX DORADO

Dorado jest tylko jednym z kilku procesorów zaproponowanych przez firmę Xerox do obliczeń symbolicznych [8, 24]. Pełen zestaw oferowanych komputerów podano w tabeli. Ostatnie dwa modele wprowadzono niedawno, aby zaspokoić zapotrzebowanie na tanie i efektywne komputery lispowe, gdyż ten fragment rynku został zdominowany przez tanie stacje robocze oraz droższe modele mikrokomputerów.

Cele projektowe, postawione przed konstruktorami tego komputera, były następujące:
- dobre parametry użytkowe, tzn. szybkość, możliwość współpracy z szybkimi urządzeniami we-wy,
- łatwość implementowania języków wysokiego poziomu,
- ograniczone rozmiary, hałas i cena.

Należy zwrócić uwagę na fakt, że Dorado (podobnie jak pozostałe maszyny Xeroxa) nie jest komputerem projektowanym tylko do Lispu.

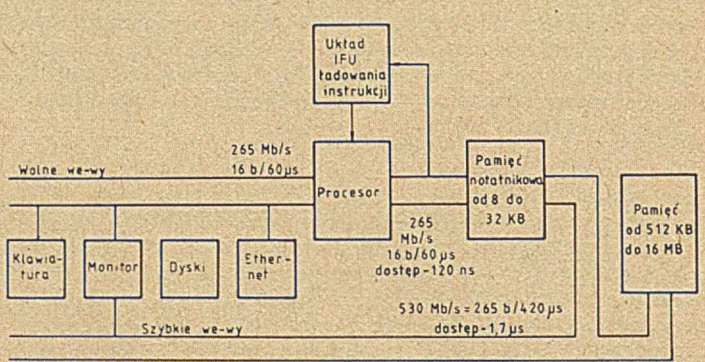
ML oferowane przez firmę Xerox

Numer	Nazwa	Orientacyjna cena (w tys. dolarów)
1100	Dolphin	45
1108	DandeLion	30
1108–111	DandeTiger	80
1132	Dorado	120
1185	?	20
1186	?	10

Podstawowe decyzje projektowe, jakie arbitralnie podjęli twórcy maszyny, można podsumować w czterech punktach:

1. Mikroprogramowany procesor, czas mikrocyklu 40 ns (udało się ostatecznie osiągnąć 60 ns), szybkość – jedna mikroinstrukcja na jeden mikrocykl.
2. Dobór odpowiednich układów pamięci spełniających wymagania narzucone przez procesor – szybka i efektywna pamięć notatnikowa, duży obszar pamięci wirtualnej.
3. Możliwość współpracy zarówno z szybkimi (np. monitory), jak i wolnymi urządzeniami we-wy.
4. Obsługa we-wy przez procesor.

Pierwszy z wymienionych punktów narzuca konkretne wymagania dotyczące szczegółów rozwiązań sprzętowych (por. rys. 5). I tak mikroinstrukcje są kolejkowane (ang. *pipelined*), dostęp do pamięci (także do notatnikowej) też jest kolejkowany, układ ładowania instrukcji (ang. *Instruction Fetch Unit*, IFU) także ma kolejkową strukturę i stanowi urządzenia autonomiczne względem procesora, a technologia wykonania jest najszybsza z osiągalnych (na początku lat osiemdziesiątych) na rynku ECL 10 k.



Rys. 5. Schemat blokowy Dorado

Języki wysokiego poziomu są implementowane przez **emulację**. Dorado jest w stanie bezpośrednio przetwarzać instrukcje specjalnego języka maszynowego, nazywanego kodem bajtowym (ang. *byte code*). Programy napisane w językach wysokiego poziomu są kompilowane na kod bajtowy (o zmiennej długości instrukcji) i następnie wykonywane. Kod operacji (pierwszy bajt kodu bajtowego) jest dekodowany przez odpowiednią do języka tablicę dekodującą i przetwarzany wyłącznie przez IFU, bez udziału procesora (z pewnymi wyjątkami).

Większa część pracy związanej z wykonaniem mikroinstrukcji przypada na IFU, procesor musi interweniować praktycznie tylko w razie wystąpienia kodów bajtowych o długościach większych niż 3 i nieudanych rozgałęzień. W większości przypadków procesor zajmuje się wyłącznie obsługą procesów we-wy widzianych jako odrębne zadania (maksimum 16 procesów o sztywno ustalonych priorytetach). Każde z zadań ma własny PC, a przełączanie zadań jest wykonywane w ramach 1 mikrocyklu. Procesor zawiera rejestry łączenia podprogramów (odrębne dla każdego zadania). Stronicowana pamięć mikroprogramu

[3] Szybka synchroniczna 32-bitowa multipleksowana magistrala, opracowana w Computer Science Laboratory MIT, a rozpropagowana przez firmę Texas Instrument. NuBus jest użyty również w ML Explorer, a także w mikrokomputerze Apple MacIntosh II.

zajmuje 4096 słów 36-bitowych i jest typu RAM, z możliwością zapisu (mikrokompilacji) przez użytkownika. Każda mikroinstrukcja podaje adres następnej instrukcji wewnątrz strony, a niektóre z mikroinstrukcji mają też wyróżnione pole podające adres innej strony. Część procesora obejmująca ścieżkę danych ma następujące zasoby i charakterystyki:

- 256 rejestrów ogólnego przeznaczenia,
- 4 stosy po 64 słowa,
- rejestry specyficzne (do sterowania stosem, do przesunięć, do arytmetyki, do obliczania adresów),
- ALU o dwóch wejściach i jednym wyjściu,
- odseparowane szyny danych: pamięci, we-wy zewnętrzną (wszystkie 16-bitowe).

Wirtualna pamięć maszyny Dorado wynosi 256 M słów (28-bitowych), a pamięć fizyczna zajmuje do 10 M słów. Odwołania do pamięci składają się z 16- lub 28-bitowego przesunięcia i 28-bitowego adresu bazowego – adres wirtualny jest ich sumą. Pamięć notatnikowa zajmuje 32 KB i składa się z 256 bloków.

Maszyny serii 1100 zostały zaprojektowane tak, by każda z nich była w stanie przetwarzać Interlisp-D. Maszyna wirtualna tego języka jest zbudowana z wykorzystaniem kodu bajtowego; operacje są implementowane częściowo w mikrokodzie, a częściowo w makrokodzie. Nie wszystkie kody operacji są dostępne w postaci mikroprogramów na każdym z procesorów – alternatywą jest mechanizm wyłapywania do standardowej funkcji Interlispu. Powyżej poziomu instrukcji wszystkie maszyny są identyczne z punktu widzenia użytkownika (między procesorami tej serii można przenosić nawet obraz pamięci).

Przy zapamiętywaniu list zastosowano znane kodowanie pola CDR [5]. Z reguły 32-bitowa komórka składa się z 24 bitów pola CAR i 8 bitów kodu CDR. Kodowanie obejmuje następujące cztery przypadki: CDR ma wartość NIL, CDR jest na tej samej stronie, CDR znajduje się w innej komórce na tej samej stronie, sama komórka jest „niewidzialnym wskaźnikiem" do pełnej (tzw. 64-bitowej) komórki na dowolnej stronie[4]).

Odzyskiwanie nieużytków jest wzorowane na GC opisanym w [7]. Tworzony jest licznik odwołań do każdego odzyskiwanego wskaźnika (miejsca w pamięci). Znajduje się on w odrębnej tablicy, a za jego zmiany odpowiada mikrokodowana funkcja, tworząca odwołanie do danej miejsca w pamięci. Odzyskiwanie, aby było to efektywne, musi się odbywać bardzo często.

Wewnętrzna architektura stosu jest wariantem modelu „stosu spaghetti" opisanego w [4]. Stos ma 128 KB.

Poniżej zamieszczono krótkie porównanie trzech starszych procesorów serii 11xx:

- 1100 (Dolphin)

4 K słowa 40-bitowej pamięci mikrokodu; czas cyklu 180 ns; IFU; mapa pamięci do 8 MB przestrzeni wirtualnej; szczyt stosu sprzętowy; 64-bitowy (do 4 słów równocześnie) dostęp do pamięci trwa ok. 2 μs.

- 1108 (DandeLion)

4 K słowa 48-bitowej pamięci mikrokodu; czas cyklu 137 ns; IFU; mapa pamięci do 16 MB przestrzeni wirtualnej; 64-bitowy (równocześnie do 4 słów) dostęp do pamięci trwa ok. 1,2 μs, dostęp bezpośredni do 16-bitowego słowa w pamięci notatnikowej 411 ns; stos w pamięci.

- 1132 (Dorado)

4 K słowa 34-bitowej szybkiej (ECL) pamięci mikrokodu; czas cyklu 64 ns; IFU; mapa pamięci do 32 MB przestrzeni wirtualnej; dostęp do 16-bitowego słowa w pamięci notatnikowej (ECL) trwa 64 ns, a przeładowanie 256 bitów tej pamięci ok. 1,8 μs.

TI EXPLORER

Explorer, opracowany w pierwszej połowie lat osiemdziesiątych przez firmę Texas Instruments (30), stanowi maszynę przeznaczoną dla jednego użytkownika do zadań związanych z tworzeniem złożonego oprogramowania, w tym symbolicznego. Podstawowe cechy Explorera:

- wyświetlacz graficzny wysokiej rozdzielczości;
- środowisko przetwarzania symbolicznego oparte na Lispie;
- rozwinięte oprogramowanie wspomagające;
- interfejs sieci Ethernet.

Komunikacja z użytkownikiem odbywa się standardowo przez monitor, klawiaturę i mysz. Dla ułatwienia wprowadzono w Explorerze rozwinięty system okien i obszerny zestaw funkcji wyjaśniających (ang. *help*) dostępnych na bieżąco. Ponieważ dostępne źródła nie podają szczegółów dotyczących architektury Explorera, można więc tylko podać jego podstawowe właściwości ujawnione przez producenta.

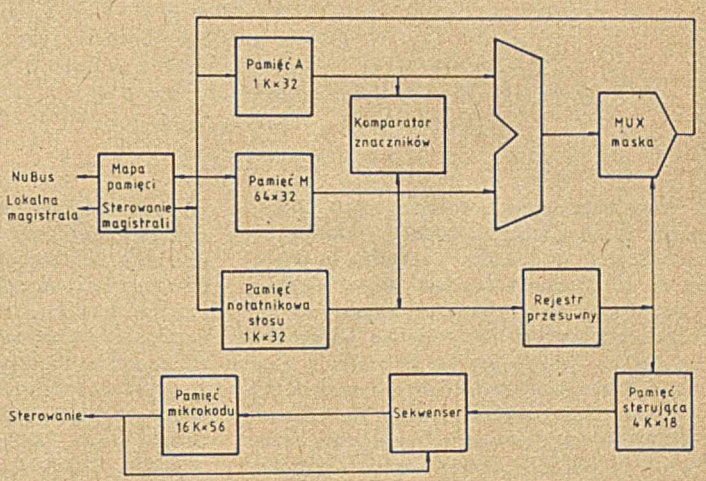
Sprzęt

- magistrala – NuBus, niezależna od procesora, wieloprocesorowa, 32-bitowa (dane, adresy), o szybkości do 37,5 MB/a;
- procesor – 1 pakiet 9U, wewnętrzna szyna danych 32-bitowa, pamięć mikrokodu – 16 K × 56 bitów, sprzętowe wspomaganie GC, przetwarzania znaczników i stronicowania, zegar mikroinstrukcji – 7 MHz;
- monitor – szybkość komunikacji z procesorem 68 Mb/s, rozdzielczość 1024 × 808 punktów obrazu;
- pamięć – dynamiczna, do 16 MB (2 pakiety 9U), o czasie dostępu 300 ns i czasie cyklu 800 ns;
- pamięć masowa – do 8 jednostek albo dysków Winchester o pojemności 112 MB, albo pamięci taśmowej (ang. *streamer*) 60 MB.

Oprogramowanie:

- Lisp – pełny Common Lisp z rozszerzeniami dającymi też pełny ZetaLisp, nakładki Flavors, Loops i Clos, duża biblioteka funkcji mikroskomplikowanych;
- środki wspomagające – edytor Zmacs, kompilator Lispu, interakcyjny interpreter (Lisp Listener), śledzenie, edytor, czcionek;
- bogate oprogramowanie graficzne;
- system menu z możliwością komunikacji w ograniczonym języku naturalnym;
- edytor tekstu klasy Scribe.

Schemat procesora Explorera przedstawiono na rys. 6.



Rys. 6. Procesor Explorera

* * *

Jak już wspomniano, istnieją interpretery i kompilatory Lispu dla praktycznie wszystkich dużych maszyn cyfrowych i dla większości mini- i mikrokomputerów. Jednak efektywność większości tych systemów pozostawia wiele do życzenia. Wynika to z nieprzystosowania klasycznych procesorów to trybu obliczeń narzucanego przez Lisp. Z reguły tradycyjny procesor jest w stanie szybko wykonać złożone operacje na słowie bądź słowach danych (dodawanie, mnożenie, operacje logiczne itd.), natomiast operacje na adresach (tzn. na słowach danych, które będą później uważane za adres) wymagają dużo wysiłku.

[4]) Taka konstrukcja jest niezbędna, aby można było wykonać operację RPLACD wówczas, gdy nie ma już wolnych komórek na stronie bieżącej.

Organizacja pamięci w klasycznych komputerach z reguły dopuszcza kilka rejestrów (szybkich) i jednorodny obszar pamięci z (niekiedy oddzielonym) jednym obszarem stosu. Lisp wymaga kilku niezależnych od siebie pamięci stosowych, wyróżnienia typu danych w słowie (tzn. długość słowa jest różna od długości pola danych), oraz położenia nacisku na operacje na adresach.

Drugim aspektem wykorzystania dotychczas istniejących komputerów jest relacja efektywności do kosztu. Na dużych komputerach maksymalna szybkość jaką dało się osiągnąć, wynosi 40 KLIPS[5] (DEC–10) i wydaje się to być granicą nieprzekraczalną (oczywiście pomijając superszybkie komputery klasy Cray). Co więcej, taka efektywność jest okupiona koniecznością przeznaczenia zasobów całego komputera (chodzi tu głównie o procesor i pamięć operacyjną) dla jednego użytkownika – Lisp jest językiem wyjątkowo pamięciochłonnym. Z kolei translatory na małe (mini- bądź mikro-) komputery osobiste nie mogą być pod względem efektywności wystarczające, choć producenci oprogramowania starają się uzyskać z nich jak najwięcej. Przykładowo, średniej klasy inerpretery Lispu dla mikrokomputerów zgodnych z IBM PC/AT są od 10 do 100 razy szybsze od znanych autorowi interpreterów dla komputerów Odra 1305 i R–32. Koszt pracy jednego użytkownika Lisp na komputerze o szybkości większej niż 10 KLIPS jest bardzo duży i praktycznie nieekonomiczne staje się używanie standardowych (niespecjalizowanych) komputerów do implementowania złożonych systemów sztucznej inteligencji.

Niewątpliwa jest natomiast potrzeba wyposażenia zarówno ośrodków naukowych, jak i badawczo-rozwojowych w maszyny Lispowe klasy stacji roboczych (tzn. jeden procesor dla jednego użytkownika). Specyfiką sztucznej inteligencji jest konieczność natychmiastowej weryfikacji hipotez badawczych, co jest praktycznie niemożliwe, a co najmniej bardzo trudne, bez możliwości użycia w każdej chwili komputera wyposażonego w Lisp mającego ponad 2 MB pamięci RAM. Co więcej, jeżeli jest tworzony system mający szansę na bezpośrednie zastosowanie w przemyśle, to jedynie szybki translator może pozwolić na osiągnięcie pożądanych wyników.

5) Miara szybkości wywodząca się z analizy efektywności systemów wnioskujących. Logical Inferences Per Second = logiczne wnioskowania na sekundę. Można przyjąć, że 1 Lisp = 1000 instrukcji maszynowych na sekundę.

KARTY PRZETWORNIKÓW ANALOGOWO-CYFROWYCH

do komputerów IBM PC/XT/AT technologia „CONVERT"; podzespoły ANALOG DEVICES

- ☐ do 32 wejść analogowych
- ☐ napięcia wejściowe: ±5V, ±10V, 0–10V, 0–20V
- ☐ dynamika 12 bitów
- ☐ przetwornica DC/DC zasilająca część analogową
- ☐ precyzyjny wzmacniacz pamiętająco-próbkujący
- ☐ dwa analogowe kanały wyjściowe
- ☐ napięcia wyjściowe: ±5V, ±10V, 0–10 V, 0–20V
- ☐ trzy 16-bitowe progromowalne liczniki
- ☐ układ generacji przerwań
- ☐ współpraca z DMA (próbkowanie 'w tle')
- ☐ licznik zdarzeń
- ☐ osiem wejściowych
  oraz osiem wyjściowych kanałów cyfrowych

DO TEGO NIEODPŁATNIE: OPROGRAMOWANIE PODSTAWOWE oraz POMOC W INSTALACJI

INFORMACJE:
'CONVERT T.S.'
53-143 Wrocław, ul. Orla 2A, Tel. 61-92-28 0/4/90

Konferencje

Informatyka i ochrona środowiska

Piąte międzynarodowe sympozjum **„Informatik für den Umweltschutz"** („Informatyka dla ochrony środowiska") odbędzie się w terminie **od 19 do 21 września br. w Wiedniu.** Organizatorami tej cyklicznej imprezy są: zachodnioniemieckie Towarzystwo Informatyczne (GI – *Gesellschaft für Informatik*), Austriacki Federalny Instytut Zdrowotności (ÖBG – *Österreichisches Bundesinstitut für Gesundheitswesen*) oraz Austriackie Towarzystwo Komputerowe (ÖCG – *Österreichische Computer Gesellschaft*).

Celem sympozjum jest zaprezentowanie i rozpropagowanie zastosowań informatyki ukierunkowanych na ochronę środowiska, a ponadto stworzenie możliwości i warunków bezpośredniego dialogu między informatykami a specjalistami działającymi na rzecz ochrony środowiska.

Tematem obrad będą metody i systemy informatyczne służące najszerzej pojętej ochronie środowiska (zachowanie klimatu i czystego powietrza, pomiary rozprzestrzeniania się zanieczyszczeń atmosfery i gleby, pomiary czystości wód, badania ekosystemów, lasów i zwierząt, ochrona przyrody, usuwanie odpadów i zanieczyszczeń, analiza i zmniejszanie uciążliwości hałasu, eliminacja zagrożeń wynikających z zastosowań energetycznych, tworzenie ogólnokrajowych i komunalnych systemów informowania o stanie środowiska, ustawodawstwo zabezpieczające skuteczną ochronę środowiska, badania systemowe i kontrola zagrożeń środowiska, związki ekologiczno-ekonomiczne, wychowanie ukierunkowane na problemy środowiska itp.).

W obszarze zainteresowań informatyki zostaną poruszone takie tematy, jak:

- tworzenie modeli i symulacja,
- stosowanie banków danych i systemów informowania,
- rejestrowanie i przetwarzanie danych dotyczących procesów,
- rozpoznawanie i przetwarzanie obrazów,
- grafika komputerowa i wizualizacja informacji,
- systemy wiedzy oraz doradcze,
- możliwości oraz ryzyko związane ze stosowaniem informatyki.

Oprócz podstawowej części konferencji, którą stanowić będzie prezentacja nadesłanych referatów i towarzysząca im dyskusja, program zostanie wzbogacony przez seminaria i referaty wprowadzające uczestników w aktualny stan zastosowań na świecie. Oprócz tego przewiduje się zorganizowanie wystawy oraz demonstrowanie najciekawszych spośród zaprezentowanych rozwiązań. Obrady przebiegać będą w trzech równoległych sesjach, w językach niemieckim i angielskim, które zostały przyjęte jako języki robocze konferencji. Materiały konferencyjne zostaną opublikowane w ramach serii *„Informatik-Fachberichte"* („Komunikaty specjalistyczne informatyki") przez wydawnictwo Springera.

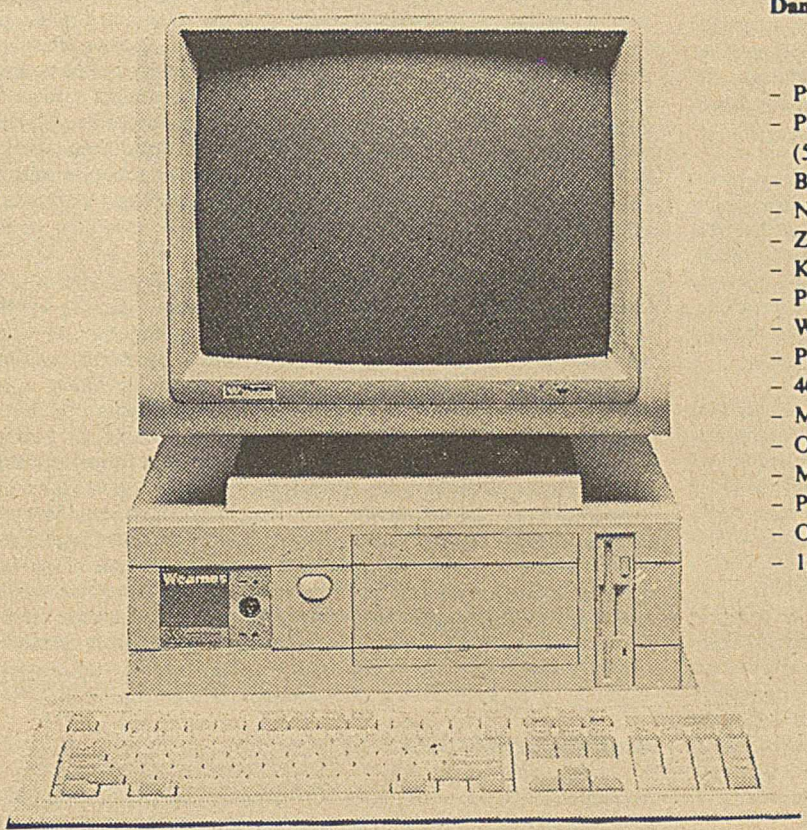
Dodatkowe informacje na temat sympozjum można otrzymać pod **adresem: dr Werner Pillmann, Österreichisches Bundesinstitut für Gesundheitswesen, Stubenring 6, A-1010 Wien, tel. (0043) 222/515 61 16.**

KOMPUTER
Z WBUDOWANĄ PRZYSZŁOŚCIĄ

Komputer zaprojektowany przez czołowego światowego producenta mikrokomputerów – firmę ADVANCED LOGIC RESEARCH i wyprodukowany przez firmę WEARNES TECHNOLOGY.

Komputer, którego możliwości i cena oszałamiają konkurencję.

Komputer, który nie zestarzeje się, dzięki możliwości rozbudowy do architektury 386SX i 486.



Dane techniczne:

SERIA WEARNES BOLDLINE „M"

- Procesor 80286 – 12.5 Mhz
- Pamięć 1 MB RAM, możliwość rozbudowy do 16 MB (5 MB na płycie głównej)
- BIOS Phoenix
- Napęd dyskietek 3,5" 1,44 MB
- Zasilacz 110 Watt
- Klawiatura 101 klawiszy
- Podstawka dla koprocesora matematycznego 80287
- Wbudowany sterownik dyskowy z przeplotem 1:1
- Port szeregowy i równoległy
- 40 MB dysk sztywny
- Możliwość korzystania z EMS 4.0
- Obudowa typu „compact"
- Miejsce na dwa napędy 5,25" o wysokości 1/2
- Podręcznik i dyskietka z programem konfiguracyjnym
- Opcjonalna rozbudowa do 386SX i 486
- 12-miesięczna gwarancja

BOLDLINE COMPUTERS
GRUPA MICOMP-TECH
Biuro Informacji Techniczno-Handlowej
ul. Astrów 7, 40–045 Katowice
telefon i telefaks: 518-628
teleks: 315687 COMP PL

DYSTRYBUTORZY:
PTH „TECHMEX"
43–300 Bielsko-Biała
ul. M. Curie Skłodowskiej 13
tel.: 42-198, 47-555, telefaks: 47-624,
teleks: 25325

EQ/1123/89

PIOTR DEMBIŃSKI
Instytut Podstaw Informatyki PAN
Warszawa

Protokoły komunikacyjne (2)

Model ISO

Model warstwowy ISO [3, 4, 5] ma siedem warstw:
- fizyczną (ang. *physical level*),
- łącza danych (ang. *data link layer*),
- sieciową (ang. *network layer*),
- transportową (ang. *transport layer*),
- sesji (ang. *session layer*),
- prezentacji (ang. *presentation layer*),
- zastosowań (ang. *application layer*).

Dlaczego akurat siedem? Liczba ta jest wynikiem uzgodnień międzynarodowych ekspertów opierających się na doświadczeniach w projektowaniu istniejących sieci, reprezentujących interesy poszczególnych producentów itd. Najogólniej rzecz biorąc, przyjęli oni założenie [8, 9], że każda warstwa powinna odpowiadać dobrze określonemu i w miarę niezależnemu zadaniu, którego wybór będzie uwzględniać istniejące międzynarodowe normy. Liczba warstw powinna być wystarczająco duża, by ważne funkcje znalazły się w osobnych warstwach, ale jednocześnie na tyle niewielka, by redukowała przepływ informacji przez sprzęgi (a więc oczywiście 7!).

Norma ISO 7498 dotyczy modelu OSI. Zawiera ona w miarę dokładną definicję pojęcia warstwy i ramową koncepcję architektury warstwowej sieci wraz z ustaleniem odpowiedniej terminologii. Ponadto zawiera opis kolejnych warstw z podaniem ich zadań i funkcji. Normalizuje konieczne usługi, które dana warstwa ma zapewnić warstwie położonej bezpośrednio nad nią w hierarchii. Krótko mówiąc, norma ISO OSI wprowadza ujednolicony podział architektury na warstwy i normalizuje nazewnictwo ich elementów składowych.

Norma ISO OSI **nie** zawiera opisu protokołów warstw. Każdy taki protokół ma osobny, już istniejący lub przygotowywany, dokument normalizacyjny, który posługuje się pojęciami i terminami ustalonymi dla danej warstwy w podstawowym modelu odniesienia. Należy również podkreślić, że sprzęgi między sąsiadującymi warstwami nie są w ogóle normalizowane. Zasada przyjęta przez ISO jest taka: norma powinna definiować jedynie „logikę" systemu, pozostawiając zupełną dowolność realizacji tej „logiki". W myśl tej zasady normy nie powinny sugerować żadnego konkretnego rozwiązania. Zagadnienie czy jest to w ogóle możliwe pozostaje dyskusyjne, ale częściowo tłumaczy, dlaczego właśnie protokoły i usługi są normalizowane, a sprzęgi nie (nawiasem mówiąc normalizacja usług napotykała i ciągle napotyka pewne opory). Jak usiłowano wskazać w przykładzie z poprzedniej części protokół jest pewną „abstrakcją" dialogu na ustalonym poziomie, który w rzeczywistości nie odbywa się bezpośrednio. Tak więc w pewnym sensie, protokół nie odnosi się w sposób prosty do konkretnych rozwiązań zapewniających ten dialog.

Trzeba też powiedzieć, że protokoły trzech pierwszych warstw (fizycznej, łącza danych i sieciowej) dotyczą sytuacji nieco bardziej złożonej niż komunikacja bezpośrednia między odpowiednimi warstwami porozumiewających się komputerów. W rzeczywistości protokoły te opisują połączenia warstwowe: komputer–węzeł komunikacyjny sieci, węzeł–węzeł oraz węzeł–komputer. Rodzaj założenia ich wszystkich tworzy właściwy protokół między komputerami. Tak więc trzy pierwsze warstwy występują zarówno w komputerach, jak i węzłach sieci, podczas gdy pozostałe tylko w komputerach.

WARSTWA FIZYCZNA

Warstwa fizyczna zajmuje się przekazywaniem w kanale komunikacyjnym sygnałów reprezentujących bity. Określa ona sposób łączenia i rozłączania fizycznych przewodników sygnałów oraz rodzaj (jednokierunkowy, dwukierunkowy) ich przesyłania podczas połączenia. Stosuje pewne zabezpieczenia (kody korekcyjne, kody detekcyjne) przeciwko błędom powstałym w wyniku zakłóceń (tzw. szumów) fizycznej transmisji sygnałów. Bardzo ważne jest dostosowanie funkcji tej warstwy do istniejących norm stosowanych w telekomunikacji. Na przykład, norma CCITT (zalecenie w nomenklaturze tej organizacji) X.21, która weszła w życie w 1976 r., określa sposób w jaki komputer abonenta nawiązuje i przerywa połączenia z urządzeniem komunikacyjnym przedsiębiorstwa telekomunikacyjnego.

WARSTWA ŁĄCZA DANYCH

Warstwa ta ma zapewnić bezbłędną transmisję, tzn. jest odpowiedzialna za rozwiązanie problemów spowodowanych gubieniem, powielaniem i uszkadzaniem wiadomości podczas przesyłania. Usługa, jaką oferuje warstwa łącza danych warstwie sieciowej, polega właśnie na gwarantowaniu – w mniejszym lub większym stopniu – niezawodności transmisji. W tym sensie usługa ta może być oferowana z parametrem jakości (i zależnej od niego ceny) do wyboru użytkownika. Warstwie łącza danych przypada na ogół także funkcja regulująca szybkość przepływu danych w wypadku nadajnika pracującego w zestawie z o wiele wolniejszym odbiornikiem (te funkcje spełniają zresztą także inne warstwy).

Warstwa łącza danych realizuje te cele umieszczając dane w „ramkach" (ang. *frames*) kontrolnych. Tak „opakowane" dane są wysyłane często wielokrotnie w celu zapewnienia poprawności odbioru. Niemal wszystkie protokoły stosowane w tej warstwie są oparte na podstawowym schemacie protokołu z przesuwnym oknem (ang. *sliding window protocol*), najprostszej sytuacji (tzn. gdy okno nadajnika i odbiornika redukuje się do jednej „klatki"), protokół ten polega na pobraniu wiadomości, utworzeniu zeń ramki danych (ang. data frame) zawierającej jej numer i identyfikujący zapamiętany jednocześnie w klatce okna i wysłaniu ramki. Następna wiadomość nie zostanie pobrana do wysyłki dopóty, dopóki nadajnik nie otrzyma ramki potwierdzającej (ang. *acknowledgment frame*). Ewentualnie, po pewnym czasie oczekiwanie na potwierdzenie, duplikat tej samej ramki zostaje ponownie wysłany. Protokół, w którym okna mają więcej klatek (liczba klatek nie musi być taka sama po stronie nadajnika i odbiornika) umożliwia przesyłanie i potwierdzanie większej liczby ramek naraz, zostawia także większą swobodę co do kolejności ich nadawania i odbierania.

WARSTWA SIECIOWA

Jak powiedziano wcześniej, opis protokołu stwarza złudzenie, jakoby komunikacja odbywała się bezpośrednio między odpowiednimi warstwami współpracujących komputerów. W rzeczywistości każdy komputer utrzymuje bezpośrednią łączność z tzw. węzłem (pod)sieci komunikacyjnej, który z kolei komunikuje się z innymi węzłami. W ten sposób jest zapewniona łączność między komputerami. Pierwsze trzy warstwy występują zarówno w komputerach, jak i węzłach. Warstwie sieciowej przypada zadanie zapewnienia połączenia komputer-węzeł. Międzynarodowy standard X.25, zaproponowany przez CCITT zawiera opis protokołu takiej łączności. Jest on powszechnie stosowany, poza krajowymi sieciami USA (w komunikacji międzynarodowej USA stosuje X.25). Innym zadaniem warstwy sieciowej jest wybór drogi między węzłami oraz ochrona sieci przed przeciążeniami i blokadami, w taki sposób, by wyższe warstwy nie musiały się kłopotać tymi problemami. Usługa warstwy sieciowej – oferowana warstwie transportowej komputera obliczeniowego – może zapewniać lub nie zapewniać połączenia wolnego od błędów zagubienia, dublowania pakietów, albo ich nadchodzenia w niespodziewanej kolejności (usługa połączenia

wirtualnego lub datagramowego). Norma X.25 dopuszcza łączność zarówno w pierwszym (do pewnego stopnia), jak i drugim przypadku. Warstwa sieciowa w sieci DECNET oferuje jedynie usługę datagramową, tzn. nie chroniącą przed wymienionymi błędami.

WARSTWA TRANSPORTOWA

Warstwa transportowa przyjmuje dane od warstwy sesji i w razie potrzeby, dzieli je na mniejsze jednostki. Przy odbiorze z warstwy sieciowej części danych składa znów w poprawną całość. Ważną funkcją tej warstwy jest niewidoczne dla warstwy sesji, tzw. zwielokrotnianie (ang. *multiplexing*), które polega na tym, że warstwa transportowa może żądać kilku połączeń sieciowych do wykonania jednego połączenia transportowego (w ten sposób zwiększa przepustowość) lub odwrotnie, kilka połączeń transportowych realizuje jednym połączeniem sieciowym (oszczędzając na połączeniach). Warstwa transportowa może przejmować także obsługę błędów, w zależności od usługi oferowanej przez warstwę sieciową, lub na skutek niewiary w jakość tych usług. Jeszcze innym zadaniem warstwy transportowej jest zarządzanie połączeniami, czyli ich zestawianie i kasowanie. Połączenia te są inicjowane i negocjowane w warstwie sesji, ale adresy użytkownika wymagają przekształcenia na adresy transportowe ujęte w pewien jednoznaczny system adresowania. Dopiero za pomocą tych adresów niższe warstwy ustalą konkretną drogę przesyłania w sieci.

Dialog wymagany do realizacji tych wszystkich zadań odbywa się bezpośrednio między warstwami transportowymi komputerów, tzn. bez pośrednictwa, jak to było w niższych warstwach, odpowiedniej warstwy węzłów. Inaczej mówięc, protokoł warstwy transportowej jest opisany w terminach wymiany PDU tej warstwy między komputerami.

WARSTWA SESJI

Jak powiedziano wcześniej, warstwa sesji inicjuje i negocjuje połączenia, tzn. żąda połączenia od warstwy transportowej, oczekuje zgody adresata i ustala z nim pewne parametry sesji, czyli rodzaj wymiany informacji w czasie trwania połączenia. Warstwa sesji sprawuje pewien nadzór nad prowadzonym dialogiem np. dba o to, by przesyłanie pewnych grup danych dokonywało się w całości) oraz dbać musi o poprawne zakończenie sesji. W wielu istniejących sieciach zadania warstwy sesji są przejmowane przez warstwę transportową; w nowszych realizacjach na ogół dba się o to, by te dwie warstwy traktować osobno, zgodnie z normą ISO.

WARSTWA PREZENTACJI

Ostateczną postacią wiadomości wysyłanej przez komputer jest ciąg bitów. Odbiorca, aby móc zrozumieć przesłaną wiadomość, musi umieć odtworzyć jej strukturę i treść z otrzymanego strumienia bitów i to niezależnie od komputera lub terminala, którego używa. Jednym słowem, nadawca z odbiorcą muszą dojść do porozumienia „w jakim języku mówią ze sobą" i jakiego systemu kodującego w tej rozmowie używają. Negocjowanie tego porozumienia to, z grubsza rzecz biorąc, zadanie warstwy prezentacji.

WARSTWA ZASTOSOWAŃ

W tej warstwie w zasadzie nie ma ograniczeń na zadania jakie mogą się okazać użyteczne w zastosowaniach. Wcześnie jednakże zauważono, że istnieją funkcje ogólne lub specyficzne, z których większość zastosowań korzysta. Skupiono się więc na definiowaniu i normalizowaniu tej grupy zadań. W niektórych realizacjach tworzą one coś w rodzaju podwarstwy, jak na przykład dość powszechnie stosowana podwarstwa CASE (ang. *Common Application Service Elements*), do której elementów odwołują się poszczególne moduły zastosowań.

Oto kilka takich „jednostek usług zastosowaniowych" (ang. *Application Service Elements*), dla których już istnieją normy międzynarodowe, lub normy te są w przygotowywaniu [7].

- ACSE (ang. *Association Control Service Element*): zawiera funkcje podstawowe do dokonania połączenia między dwoma procesami niezależnie od ich lokalizacji;
- CCR (ang. *Commitment, Concurrency and Recovery*) o funkcjach umożliwiających utrzymanie integralności rozproszonej bazy danych;
- RTS (ang. *Reliable Transfer Server*), jego funkcje zapewniają ciągłość komunikacji we współpracy z warstwą sesji;
- ROS (ang. *Remote Operation Service*) o funkcjach koniecznych do tego, by moduły zastosowań mogły komunikować się z ich użytkownikami;
- MHS (ang. *Message Handling System*), jego funkcje realizują zadania poczty elektronicznej poza samym przesyłaniem,
- FTAM (ang. *File Transfer, Access and Management*) o funkcjach koniecznych do przesyłania plików i przekształcania ich na odległość;
- VT (ang. *Virtual Terminal*), jego funkcje umożliwiają współpracę dowolnego terminala z dowolnym komputerem sieci.
- TP (ang. *Transaction Processing*), jego funkcje „porządkują" proces wykonywania operacji na bazach danych rozproszonych i powielanych, tak aby w wyniku tych operacji baza pozostawała w stanie spójnym.
- ODA (ang. *Office Document Architecture*) o funkcjach umożliwiających tworzenie, formatowanie, zmianę i przesyłanie dokumentów na odległość.

Podsumowując można zapytać, jaki jest stan zaawansowania prac normalizacyjnych w protokołach sieciowych i ich wdrożeń. Cały proces normalizacyjny rozpoczął się na dobre w 1978 r. i można przewidywać, że zakończy się około 1998 r. [7]. Znajdujemy się więc w połowie drogi. Protokoły warstw 1–4 są już dobrze ustabilizowane w istniejących sieciach. Protokoły wyższych warstw są wprowadzane stopniowo. Należy podkreślić, że wdrożenie protokołu i jego stabilizacja trwa od pięciu do dziesięciu lat, a nowe protokoły pojawiają się sukcesywnie.

Normy protokołów starają się zaspokoić największą liczbę wymagań i oferują pełną gamę możliwych usług, ich parametrów i atrybutów w ramach postawionego zadania. Formułują one jednocześnie pewien podstawowy zbiór tych usług, konieczny w każdej realizacji, pozostałe traktując jako fakultatywne, tzn. takie, z których można zrezygnować w konkretnej implementacji, w zależności od przewidywanych potrzeb i ograniczeń. Każdy producent lub grupa producentów definiuje własne tzw. profile (ang. *profiles*) standardów, czyli normy, w których dokładnie wymienia się zrealizowane elementy opcjonalne. Przykładami takich grup producentów przygotowujących profile obowiązujące dla danej grupy są: amerykańska korporacja COS (*Corporation for Open Systems*), organizmy MAP i TOP (także amerykańskie), z których pierwszy zajmuje się zastosowaniami przemysłowymi, a drugi naukowymi i biurowymi, japońska organizacja POSI (*Japonese Promoting Conference for OSI*), wreszcie grupa SPAG (*Stadards Promotion and Application Group*) zrzeszająca 12 największych konstruktorów europejskich (ICL, Siemens, Thompson, Philips, Bull, Olivetti itd.). Prace tych grup mają też ogromny wpływ na ustalanie samych norm.

FORMALNE TECHNIKI SPECYFIKACJI PROTOKOŁÓW

Normy międzynarodowe są formułowane z reguły w języku angielskim (amerykańskim) z użyciem pomocniczych środków, takich jak tablice, diagramy itp. Taki sposób opisu skomplikowanych systemów prowadzi do wielu niejasności i zostawia miejsce na różne interpretacje. Nie pozwala także na stworzenie i zastosowanie wspomagającego oprogramowania, mogącego usprawnić projektowanie i powstawanie opisu protokołu, jego weryfikację i wreszcie implementację. Z tych powodów we wczesnych latach osiemdziesiątych rozpoczęto w ISO prace nad opracowaniem formalnych technik (języków), w których opisy protokołów nie miałyby wymienionych wad opisów w językach naturalnych. W wyniku tych prac powstały dwa sformalizowane języki specyfikacji (FDTs – *Formal Description Techniques*): Estelle [2] i Lotos [1]. Oba uzyskały już (1988) status standardu międzynarodowego IS (ang. *International Standard*). Wcześniej normą (zaleceniem) CCITT stał się język SDL (ang. *Specification and Description Language*) [3]. Estelle i SDL są oparte na modelu kooperujących automatów, a Lotos – na algebraicznym podejściu do opisu współpracy procesów zaproponowanym w [4]. Charakteryzując bardzo pobieżnie te trzy techniki opisu można powiedzieć, że mocną stroną SDL jest jego sformalizowana reprezentacja graficzna. Estelle daje najwięcej możliwości szybkiego uzyskania prototypowych implementacji, a Lotos wyróżnia się eleganckimi mechanizmami przekształcania opisów na różne poziomy abstrakcji.

W tych trzech językach powstały dotychczas specyfikacje wielu protokołów (np. warstwy sieciowej, transportowej, sesji, a także protokoły warstwy zastosowań FTAM, VT i inne). Specyfikacje te nie zastępują oficjalnych norm sformułowanych w języku naturalnym, ale ewentualnie je uzupełniają (co też nie przebiega bez oporów).

Bardzo ważnym elementem, który zapewne zadecyduje o powodzeniu w rozpowszechnianiu tych formalnych metod opisu – być może nie tylko

dla protokołów – będzie skuteczność i łatwość w użyciu rozlicznych narzędzi programistycznych (edytorów, kompilatorów, symulatorów) ułatwiających projektowanie, weryfikację i implementację powstających w nich opisów systemów. Takie narzędzia powstały już w wielu ośrodkach i są w stadium eksperymentalnych zastosowań [4].

Protokoły warstwy zastosowań mają najczęściej do przekazania złożone informacje (często o bardzo szczególnym charakterze), których zdefiniowanie wymaga pewnej strukturalizacji i standaryzacji. Te złożone informacje są przekazywane do warstwy prezentacji, w której są kodowane w ciągi bitów. Od pewnego czasu narastało przekonanie, że normy protokołów w warstwie zastosowań muszą posługiwać się jakimś spójnym i jednolitym systemem definiowania (typów) danych i jednocześnie odwoływać się do pewnego „uniwersalnego" systemu ich kodowania. Bez takich wspólnych ram trudno sobie wyobrazić spójność opisów w różnych zastosowaniach, a także zgodność realizacji opartych na tych opisach.

Wynikiem tego przekonania było powstanie dwóch, ściśle ze sobą związanych standardów ISO/CCITT: ASN. 1 (*Abstract Syntax Notation One*) [5] i Bazowe Reguły Kodowania (*Basic Encoding Rules*) [6]. Pierwszy z nich definiuje składnię (i do pewnego stopnia semantykę) języka definiowania typów danych oraz metodologię jego używania. Drugi formułuje jednoznaczny przepis kodowania (i pośrednio dekodowania) informacji i jej struktury (typu) zadanej w ASN. 1. Oba standardy są pewnymi, nieznacznie zmodyfikowanymi, wersjami zaleceń zawartych w dokumencie X.409, opracowanym wcześniej przez CCITT. Obie normy zostały bardzo szybko zaakceptowane przez środowisko i cokolwiek o nich można powiedzieć nie zmienia to faktu, że już są i z całą pewnością będą powszechnie stosowane w definiowaniu protokołów nowej generacji. Wiele firm opracowało już – lepsze lub gorsze – kompilatory, które tłumaczą struktury ASN. 1 na struktury danych wybranego języka programowania (głównie języka C, ale także Pascala lub Cobolu) i na odwrót oraz produkują procedury automatycznego kodowania i dekodowania zgodne z Bazowymi Regułami Kodowania.

Można postawić w tym miejscu pytanie, dlaczego w językach takich jak Estelle, Lotos czy SDL nie zastosowano mechanizmów ASN. 1 definiowania typów danych. Takie połączenie dawałoby bowiem ujednolicone i całościowe formalizmy specyfikacji. Złożyły się na to przyczyny natury historycznej, organizacyjnej (wewnątrz ISO i CCITT) i w pewnym stopniu, technicznej (silne ukierunkowanie ASN. 1 i specyficzny rodzaj jego interpretacji). Współistnienie dwóch formalizmów w jednym opisie stwarza pewne problemy, które jednak, do celów praktycznego korzystania z takich opisów, udaje się częściowo rozwiązać za pomocą, na przykład, zabiegów podobnych do tych, jakie stosuje się przy rozdzielnej kompilacji (ang. *separate compilation*).

Implementacja norm protokołów nie może tylko polegać na wierze w zdolności programistów do poprawnego zrealizowania zawartych w tych normach zaleceń. Wymaga dokładnej i wielostronnej weryfikacji. Jakkolwiek zaawansowane byłyby metody projektowania i sprawdzania poprawności programów w procesie ich powstawania, nie można uniknąć konieczności testowania zgodności (ang. *conformance testing*) systemu z jego specyfikacją (normą) już po jego uruchomieniu. Pytanie polega na tym, co i jak testować oraz jak zagwarantować jednolitość i spójność przeprowadzanych testów w różnych realizacjach. Pytania to nie łatwe i opracowanie odpowiedzi na nie wymaga znowu wspólnego, międzynarodowego wysiłku. Podjęto go i tym razem, w 1984 r. w ISO w ścisłej współpracy z CCITT. Punktem wyjściowym rozpoczętych prac było rozróżnienie dwu ich zasadniczych celów: znormalizowania metodologii testowania protokołów i uzyskania norm ustalających zbiory („gry") testów dla poszczególnych protokołów. Metodologię testowania ujęto w pięciu częściach, z czego dwie pierwsze osiągnęły status Wstępnych Propozycji (ang. *Draft Proposals*), a inne są w stadium opracowywania roboczych dokumentów. Gry testów dla poszczególnych protokołów są przedmiotem intensywnych badań dyskusji prowadzonych wewnątrz firm komputerowych, w ich zgrupowaniach (COS, POSI, SPAG i inne), wreszcie w krajowych organizacjach normalizacyjnych.

Jednym z problemów metodologii testowania był wybór odpowiedniego formalizmu, w którym znormalizowane gry testów byłyby zdefiniowane. Kandydatami były języki Estelle, Lotos i SDL, ale zdecydowano się raczej (decyzja nie jest ciągle ostateczna) na opracowanie nowej wyspecjalizowanej notacji nazwanej TTCN (ang. *Tree and Tabulag Combined Notation*). W dużym uproszczeniu jest to język tablic, w których w jednej z kolumn jest przedstawiona dynamika zdarzeń testujących (interakcji wejściowych i wyjściowych) w postaci drzewa wystąpień tych zdarzeń. W innych kolumnach, z każdym wystąpieniem zdarzenia w drzewie są stowarzyszone jego pewne atrybuty, parametry stałe, a także możliwy werdykt testowania: zawiódł (ang. *failed*), przeszedł (ang. *passed*), nierozstrzygnięty (ang. *inconclusive*). Choć TTCN nie jest jeszcze językiem zatwierdzonym, a nawet do końca zdefiniowanym, wiele próbnych specyfikacji systemów testów zostało już w nim zaproponowanych. Nie jest to, w zasadzie, język bezpośrednio implementowalny, ale język przeznaczony do możliwie jednoznacznego opisu zbioru sekwencji testujących dla określonego standardu protokołu OSI lub jego wybranego profilu.

* * *

Istnieje niezliczenie wiele innych propozycji formalizmów służących specyfikacji i projektowaniu protokołów komunikacyjnych, nie mówiąc już o językach programowania, w których ostatecznie są one implementowane. W powyższym przeglądzie skupiono się tylko na tych, które przeszły, lub przechodzą proces normalizacji równoległy do procesu, któremu podlegają same protokoły. Wydaje się, że takie podejście pozwala spojrzeć na rozległą i skomplikowaną dziedzinę z perspektywy unifikującej często na pozór rozbieżne tendencje.

LITERATURA

[1] Bolognesi T., Brinksma E.: Introduction to the ISO Specification Language Lotos. Computer Networks and ISDN Systems. No. 14, 1987

[2] Budkowski S., Dembiński P.: An Introduction to Estelle: a specification language for distributed systems. Computer Networks and ISDN Systems, 1988 (także Materiały Jesiennej Szkoły PTI, Mrągowo 1987)

[3] CCITT SG XI, Recommendation Z. 100. 1987

[4] Diaz M., Vissers Ch., Budkowski S.: Estelle and Lotos Software Environments for the Design of Distributed Systems. Proc. ESPRIT Conference Week, 1987

[5] ISO 8824, Information Processing Systems. Open Systems Interconnection, Specification of Abstract Syntax Notation One. 1987

[6] ISO 8825, Information Processing Systems, Open Systems Interconnection.

[7] Pujolle G.: L'etat de normalisation de la couche application OSI. Proc. CFIP'88, Eyrolles, Paris 1988

[8] Tanenbaum A.S.: Sieci komputerowe. WNT, Warszawa 1988

[9] Zimmermann H.: OSI Reference Model – The ISO Model of Architecture for Open Systems Interconnection, IEEE Trans. on Comm., Vol. COM-28, 1980.

Przypominamy!
Do końca sierpnia można jeszcze zamówić prenumeratę INFORMATYKI na czwarty kwartał.

CZĘŚCI ZAMIENNE DO URZĄDZEŃ

MEMOREX

Skład Celny – Urządzenia elektroniczne
PP„ZETO Poznań"

ul. Fredry 8a
60-967 Poznań

tel. 52-92-21
teleks 0413380

ZO/33/90

MEDCOM

ZASILANIE AWARYJNE

ZASILACZE BEZPRZERWOWE

naszej produkcji

UPS–200
UPS–300

produkcji tajwańskiej

UPS–500

produkcji japońskiej

GENERATORY PRĄDOTWÓRCZE

różnych mocy

EO/3/90

Biuro
Al. Ujazdowskie 26/39
00–478 Warszawa tel. 28 93 57, teleks 81 70 60

ZAKŁAD PRODUKCYJNO-USŁUGOWY
ul. Wołodyjowskiego 41
02–724 Warszawa tel.43 04 41

Ada dla mikrokomputerów MacIntosh

Firma Meridian, mająca w swoim dorobku implementacje kompilatorów Ady dla mikrokomputerów IBM PC oraz System 8000 firmy Zilog, zaprezentowała kompilator Ada Vantage dla mikrokomputera MacIntosh. Kompilator kosztuje 1195 USD, pracuje w środowisku programowym MacPro i wykorzystuje jako kompilator MPW C (kod źródłowy Ady najpierw jest tłumaczony na kod w języku C). Podobnie jak inne kompilatory Ady firmy Meridian pracujące pod kontrolą systemów MS-DOS, PC-DOS i Unix, również wersja 2.2 omawianego kompilatora uzyskała oficjalny atest.

Do kompilatora są dostarczane dodatkowe biblioteki i programy usługowe uzupełniające podstawowy zestaw kompilatora, np. pakiet implementujący bibliotekę matematyczną i program uruchomieniowy.

TYPY DANYCH

W kompilatorze zaimplementowano całkowitoliczbowe i zmiennoprzecinkowe typy danych. Dane krótkie całkowitoliczbowe *(ang. short integer)* zajmują 2 bajty, całkowitoliczbowe – 4 bajty, dane zmiennoprzecinkowe wymagają 8 bajtów pamięci, a stałoprzecinkowe długie i krótkie odpowiednio – 2 i 4 bajty.

W omawianej wersji 2.2 kompilatora nie zaimplementowano dostępnych w mikroprocesorze MC68881 liczb zmiennoprzecinkowych o reprezentacji 12-bajtowej. W instrukcji użytkownika podano, że implementacja obejmuje dwa typy stałoprzecinkowe, reprezentowane za pomocą liczb całkowitych o urojonej części dziesiętnej. Ze złożonych struktur danych są dostępne rekordy z wyróżnikami (deklaracja rekordu zawiera argumenty pozwalające na określenie w podprogramie aktualnej wielkości rekordu) oraz tablice o zmiennym rozmiarze, tj. o wymiarach zdefiniowanych jako argumenty; wymiar tablicy jest nieokreślony do chwili zdeklarowania zmiennej typu tablicowego.

Mimo, że implementacja AdaVantage zawiera pełny pakiet *text_io* (pierwotnie zdefiniowaną jednostkę biblioteczną z procedurami obsługującymi plikowe operacje we-wy), to firma Meridian dostarcza dodatkowe trzy prostsze biblioteki we-wy: *iio*, *fio* i *ada_io*. Pakiet *iio* jest konkretyzacją *text_io.integer_io* za pomocą prostej instrukcji Ady:

```
With text_io; use text_io;
package iio is new
text_io.integer_io(integer);
```

Podobnie, pakiet *fio* jest konkretyzacją *text_io.float_io* dla typów zmiennoprzecinkowych. Pakiet *ada_io* zawiera bibliotekę podstawowych procedur obsługi we-wy z konsoli dla znaków, napisów, liczb stało- i zmiennoprzecinkowych. Ponieważ *iio*, *fio* i *ada_io* nie są częścią pakietów standardowych, ich dołączenie umożliwia pełne zunifikowanie tworzonych programów.

PLIKI

W implementacji AdaVantage pliki mogą być traktowane jako tablice bajtów (z wykorzystaniem *direct_io*) lub jako strumień bajtów *(sequential_io* lub *text_io)*. Jednocześnie można otworzyć nie więcej niż 30 plików; ograniczeniem może być rozmiar wolnej pamięci. Nazwy plików i skorowidzów muszą być zgodne z regułami systemu operacyjnego MacIntosha. Oczywiście stwarza to problemy przy przenoszeniu kodu źródłowego w środowisko innego systemu operacyjnego – może wyniknąć konieczność przeredagowania stałych napisowych reprezentujących nazwy plików.

Programy wykonywane w środowisku MPW mogą korzystać z mechanizmów potoków i przeadresowania we-wy. Na przykład, polecenie *Ada_lister|sort* powoduje przesyłanie wyników programu napisanego w Adzie *(Ada_lister)* do programu sortującego. W środowisku MPW istnieje kilka pierwotnie zdefiniowanych urządzeń specjalnych, np. konsola, puste urządzenie wyjściowe, standardowe wyjście i wejście strumieniowe, jak również wyjście diagnostyczne.

SZCZEGÓŁY IMPLEMENTACJI

W kompilatorze AdaVantage jednostki i typy rodzajowe zaimplementowano jako makra. W tym wypadku istnieje możliwość wyboru między zwiększeniem szybkości wykonania a zwiększeniem objętości generowanego kodu rodzajowego. W celu zaoszczędzenia pamięci firma zaleca stosowanie z umiarem konkretyzacji kodu rodzajowego. Wykorzystanie rodzajowości jest możliwe wówczas, gdy ciało kodu rodzajowego i jego specyfikacja znajdują się w tym samym pliku.

Kompilator umożliwia łączenie programu źródłowego Ady z podprogramami w języku C i języku asemblera. Przez pragmę (tj. dyrektywę kompilatora) określa się język, nazwę importowanego podprogramu i jego opcjonalną nazwę do konsolidacji. Można, na przykład, napisać w implementacji MPW języka C edytor wierszowy o nazwie *oneLineEdit* (wydruk).

W kompilatorze AdaVantage w ograniczony sposób zaimplementowano wielozadaniowość. Przy przełączaniu zadań nie wykorzystuje się podziału czasu z wywłaszczeniem. Zamiast tego procesor przechodzi od zadania do zadania w punktach aktywacji, wejść, zakończenia i w stanach oczekiwania. Z tego powodu, długo wykonywane procedury (zawierające pętle) nie uwzględniające wspomnianych mechanizmów przełączania procesora, powinny być specjalnie traktowane ze względu na niebezpieczeństwo zablokowania innych zadań. W instrukcji użytkownika sugeruje się,

```
/* MPW C */
void oneLineEdit(s)
char *s
{
  ...
}

--AdaVantage
package editLine is
  procedure oneLineEdit(s: string);
     pragma interface(c, oneLineEdit)
end editLine;

with editLine;
procedure get_line(s: string)
  is
begin
  editLine.oneLineEdit(s);
end get_line;
```

Wydruk

aby – na przykład – pętle nieskończone zawierały instrukcję *delay 0.0*, pozwalającą przeszeregować bieżące zadanie.

Pragma *priority* powoduje przypisywanie priorytetu zadania podczas kompilacji. Ponieważ norma Ady nie określa domyślnych priorytetów zadań, więc w omawianym kompilatorze przyjmuje się prioryter najniższy. Każdemu zadaniu przydzielono stos o wielkości 2 KB, ale programista może ją zmienić.

KORZYSTANIE Z KOMPILATORA

AdaVantage umożliwia kompilowanie znajdujących się w oddzielnych plikach samodzielnych programów, specyfikacji pakietów lub ich treści i podprogramów. Kompilowane pliki są rozróżniane na podstawie rozszerzenia nazwy, w znaczeniu systemu MS-DOS.

Istnieje kilka dyrektyw sterujących globalnymi krokami kompilacji. Jedna z nich pozwala kompilować źródłowy program Ady na kod źródłowy MPW C. Za pomocą innej, można do kodu w języku C dołączyć jako komentarze kod Ady. Są również dostępne dyrektywy kontroli wykonania programu, przydatne na etapie testowania i uruchomiania programu. Dyrektywa sprawdzania operacji numerycznych pozwala wychwycić dzielenie przez zero i wystąpienie przepełnienia. Ogólna kontrola obejmuje oprócz tego sprawdzanie zakresów tablic, długości zmiennych napisowych itp. Wyłączenie sprawdzeń zmniejsza objętość wykonywalnego kodu i zwiększa szybkość wykonania.

Inna grupa dyrektyw kompilatora odnosi się do uruchomiania. Dyrektywa *-fD* umożliwia śledzenie skompilowanego kodu za pomocą samodzielnego programu uruchomieniowego Meridian AdaVantage Debugger.

Dyrektywa *-g* pozwala wykorzystać maszynowy program uruchomieniowy MacsBug (zostaje ona przeniesiona do kompilatora MPW C i powoduje umieszczenie nazwy każdej procedury w generowanym kodzie wynikowym).

Program optymalizujący jest częścią kompilatora. Za pomocą dyrektyw kompilatora i konsolidatora można optymalizować objętość generowanego kodu lub szybkość wykonania. Przy optymalizacji objętości, kompilator odrzuca nie wykorzystane części kodu importowane z bibliotek. Szybkość wykonania można zwiększyć przez wyłączenie kontroli, a w przypadku mikrokomputera MacIntosh II wyposażonego w procesor MC68020 – przez spowodowanie wytwarzania jego rozkazów.

Firma Meridian dostarcza program *ada/f* – specjalną wersję kompilatora, dostosowaną do kompilacji programów zawierających szczególnie dużo błędów. Komunikaty o błędach mogą zostać przekazane potokowo do programu *ff*, który je przetwarza na wykonywalne polecenia edytora. Polecenia te można wykorzystać do zlokalizowania i poprawienia odpowiednich wierszy programu w odpowiednim pliku.

Polecenie BAMP (ang. Build Ada Main Program) wywołuje konsolidator MPW oraz generuje programy wykonywalne w Mac Finder lub w powłoce MPW. BAMP domyślnie tworzy program pracujący tylko w środowisku MPW. Zmieniając dyrektywy można wyjść poza segment 32 KB, wykonać globalną i lokalną optymalizację, wybrać inną bibliotekę czasu wykonania oraz zażądać wydruku mapy symboli.

Istnienie wzajemnych powiązań między bibliotekami i skompilowanymi jednostkami narzuca korzystanie z bazy danych biblioteki. Pliki tej bazy umożliwiają nadzorowanie przez kompilator integralności i poprawności wersji kodu. Pakiet zawiera pewną liczbę programów usługowych do tworzenia, pielęgnowania, modyfikowania i usuwania informacji znajdujących się w bazie danych biblioteki.

Kompilator, BAMP, zarządzanie biblioteką i zawsze dostępny samouczek są wywoływane w rozwijalnym menu wykorzystującym standardowe rozwiązania MacIntosha. Na ekranie znajdują się przełączniki opcji, rozkazów, wewnętrznie tworzone okna do podawania poleceń i okna samouczka. Przełączniki opcji pozwalają określić wszystkie parametry pracy. Po naprowadzeniu myszką kursora na prostokąt oznaczający opcję rozkazu, naciśnięcie guzika powoduje przełączenie trybu aktywności polecenia, modyfikatora, opcji itd. Jeśli guzik jest naciśnięty dłużej, to w oknie pomocniczym zostaje wyświetlony krótki opis polecenia. Po wybraniu opcji, pliku lub bibliotek okno rozkazowe jest automatycznie uaktualniane.

TESTY PORÓWNAWCZE

Programy do testowania są oparte na często wykonywanych działaniach, takich jak sortowanie lub odwracanie macierzy. Pierwsze trzy testy to: *Sito* (Eratostenesa), *Sortowanie* – test polegający na utworzeniu uporządkowanej tablicy zawierającej 1000 liczb całkowitych i odwrotnym uporządkowaniu metodą Shella-Metznera, *Zmiennoprzecinkowy* – standardowy test miesięcznika BYTE, w którym wykonuje się 5000 razy podstawowe działania. W teście *Odwracanie Macierzy* jest tworzona macierz 20 × 20 i wszystkim elementom diagonalnym przypisuje się liczbę 2, a pozostałym liczbę 1, pomiarom podlega czas odwrócenia tak utworzonej macierzy.

Tabela 1. Wielkości plików

Test	Wielkość pliku (KB)		
	Źródłowego	Nieoptymalizowanego	Optymalizowanego
Sito	1	22	21
Sortowanie	3	26	24
Zmiennoprzecinkowy	1	30	28
Odwracanie	2	31	29
Rekurencja	2	23	21
Zapis	1	28	27
Odczyt	1	27	26

W teście *Rekurencja* wykorzystuje się rekurencyjną wersję popularnego algorytmu szybkiego sortowania.

Dwa ostatnie testy dotyczą operacji dyskowych: są to: *Odczyt* i *Zapis*, standardowe testy BYTE mierzące czas odczytania i zapisania pliku na sformatowaną, dwustronną 3,5-calową dyskietkę o podwójnej gęstości, umieszczoną w wewnętrznym napędzie mikrokomputera.

Tabela 2. Czasy kompilacji i konsolidacji

Test	Czas kompilacji i konsolidacji (min : s)	
	Nieoptymalizowany	Optymalizowany
Sito	3:39	3:43
Sortowanie	4:13	4:12
Zmiennoprzecinkowy	4:17	4:14
Odwracanie	4:28	4:23
Rekurencja	4:04	4:12
Zapis	3:32	3:30
Odczyt	3:26	3:28

Testy porównawcze były wykonywane na mikrokomputerze MacPlus wyposażonym w dysk o pojemności 20 MB, firmy DataFrame. Były uruchamiane w środowisku MPW. W tabelach 1–3 umieszczono wyniki testów dla dwóch wersji – z optymalizacją i bez. Czasy kompilacji i konsolidacji dla wszystkich programów testowych były raczej duże, ze względu na to, że kompilator Ady najpierw tłumaczy kod Ady na kod w języku C. Dla wersji optymalizowanych wyłączono wszystkie typy sprawdzeń, co spowodowało znaczne poprawienie czasów wykonania testów *Sito*, *Sortowanie*, *Odwracanie Macierzy* i *Rekurencja*. Zmiana dotycząca pozostałych nie była tak znacząca.

Tabela 3. Czasy wykonania

Test	Czas wykonania (min : s)	
	Optymalizowany	Nieoptymalizowany
Sito	0:34	0:08
Sortowanie	2:36	0:18
Zmiennoprzecinkowy	1:08	1:08
Odwracanie	0:39	0:27
Rekurencja	0:44	0:11
Zapis	0:19	0:19
Odczyt	0:13	0:13

* * *

W artykule omówiono pierwszy handlowy kompilator Ady dla mikrokomputerów MacIntosh. Podczas gdy czasy wykonania programów testowych można ocenić jako dobre lub zadowalające, cykl kompilacji i konsolidacji jest bardzo długi. Cena pakietu sugeruje, że AdaVantage jest przeznaczony dla programistów profesjonalnych. Stosunek ceny do efektywności można oszacować jako nieznacznie niekorzystny dla implementacji, zwłaszcza że do ceny należy doliczyć cenę powłoki MPW.

Należy oczekiwać, że profesjonalni programiści Ady wybiorą implementacje dostępne dla mikrokomputerów IBM PC, w tym również firmy Meridian o cenie poniżej 1000 dolarów.

Oprac. **M. KUC**
Na podst. BYTE

Wychodząc naprzeciw zapotrzebowaniu środowiska, redakcja nasza oferuje łamy INFORMATYKI na bezpłatne zamieszczanie anonsów o imprezach informatycznych organizowanych na terenie całego kraju.

Zwracamy się do organizatorów konferencji, wystaw, konkursów i sympozjów z prośbą o przysyłanie do redakcji krótkich (pół strony maszynopisu) informacji o planowanych imprezach. Teksty powinny zawierać datę i miejsce imprezy, krótką charakterystykę jej tematyki oraz adres organizatorów.

Oczekujemy na zgłoszenia.

Informatyka na wyższych uczelniach RFN

Mimo sukcesywnego wzrostu bezrobocia wśród absolwentów wyższych uczelni w RFN, informatyka należy tam do tych wyjątkowych kierunków specjalizacji zawodowej, gdzie zjawisko to nie występuje. Według danych za I kwartał br. w RFN było 30–40 tys. wolnych miejsc pracy dla informatyków. Nic więc dziwnego, że dla absolwentów szkół średnich studia informatyczne pod względem atrakcyjności zajmują od wielu już lat czwarte miejsce (po budowie maszyn, ekonomii i elektrotechnice). Gdyby nie fakt, że są to studia bardzo trudne, pozycja ta była by jeszcze lepsza.

W semestrze zimowym 1988/1989 na wyższych uczelniach RFN informatykę jako specjalizację główną studiowało ok. 33 000 osób. Rok później o przyjęcie na ten kierunek studiów ubiegało się 7593 kandydatów. Teoretycznie każdy z nich miał szansę przyjęcia w jednej z uczelni na terenie kraju, ponieważ ze względu na wspomniany już popyt na wysoko wykwalifikowanych informatyków na kierunku tym nie obowiązuje tzw. *numerus clausus*, tzn. rygorystyczne limitowanie liczby osób przyjmowanych w każdym roku na dany kierunek studiów. We wspomnianym roku ostatecznie przyjęto tylko 3936 osób, ze względu na ograniczone warunki lokalowe uczelni, a zwłaszcza brak ok. 100 pracowników dydaktycznych.

Trudny program studiów informatycznych w połączeniu z ostrą kontrolą postępów w ciągu pierwszych dwóch lat nauki powodują, że dyplom uzyskuje nie więcej niż 40% osób rozpoczynających te studia. W ten sposób w 1989 r. pracę zawodową rozpoczęło zaledwie 1200–1500 dyplomowanych informatyków (niem. *Diplom-Informatiker*), co oczywiście w minimalnym stopniu wpłynęło na złagodzenie wspomnianego na wstępie zapotrzebowania.

Ponieważ tak atrakcyjne pod względem perspektyw zatrudnienia studia nęcą wiele osób nie posiadających odpowiednich uzdolnień i predyspozycji psychicznych (uzdolnienia matematyczne, logiczno-analityczny sposób rozumowania, dokładność, a nawet pewne cechy twórcze), tak popularni w RFN doradcy wyboru zawodu (niem. *Berufsberater*) sugerują zainteresowanym studiami informatycznymi uprzednie poddanie się bezpłatnym badaniom psychotechnicznym, jakie oferuje młodzieży każdy miejscowy urząd pośrednictwa pracy. Może to zaoszczędzić wielu rozczarowań, a zwłaszcza straty cennego czasu, jaki trzeba poświęcić na przygotowanie się do samodzielnego życia.

Osoby przekonane już o posiadaniu właściwych uzdolnień i predyspozycji muszą teraz zdecydować się nie tylko na wybór uczelni, ale i kierunku specjalizacji informatycznej. Możliwości wyboru są szczególnie duże. Stwarzają je aż 42 uczelnie w 36 miastach, w których można studiować informatykę jako specjalizację podstawową nazywaną tam *„Allgemeine Informatik"* (kierunek ogólnoinformatyczny).

Jeśli wierzyć badaniom ankietowym przeprowadzonym wśród studentów informatyki, najlepsze warunki studiów istnieją na uniwersytetach w Saarbrücken, Erlangen-Norymberdze oraz Clausthal, natomiast najniższe lokaty spośród 20 uczelni uzyskały wyższe szkoły techniczne w Kilonii, Brunszwiku i Berlinie.

Oprócz studiów czysto informatycznych wiele uczelni oferuje studia ukierunkowane na określoną specjalizację informatyczną, taką jak informatyka inżynierska (Dortmund) lub informatyka gospodarcza (Bamberg, Darmsztat, Mannheim). Główną specjalizacją zawodową może być np. informatyka stosowana, ukierunkowana na socjologię lub lingwistykę, jaką oferuje wyższa szkoła pedagogiczna w Koblencji. Od 1972 r. wydziały medycyny uniwersytetu w Heidelbergu oraz wyższej szkoły zawodowej w Heilbronn oferują specjalizację „informatyka medyczna" (niem. *medizinische Informatik*). Coraz częściej informatyka staje się dominującym przedmiotem również na innych kierunkach studiów, takich jak matematyka, elektrotechnika, ekonomia lub prawo. Ilustracją takich tendencji może być program wykładów i zajęć w Uniwersytecie Tübingen, gdzie w semestrze zimowym 1989–1990 położono bardzo duży nacisk na informatykę aż na 10 wydziałach, specyfikując w ich programach nauczania łącznie 76 pozycji bezpośrednio związanych z tym przedmiotem. Przykładowo, na wydziale filozofii zauważyć można „Automatyzację dowodu", na wydziale historii sztuki – „Ilościową etnologię", a na wydziale biologii – „Genetykę komputerową".

Studia na kierunku ogólnoinformatycznym są podzielone na studia podstawowe (niem. *Grundstudium*), kończące się dyplomem I. stopnia (niem. *Vordiplom*) oraz studia zasadnicze (niem. *Hauptstudium*), uwieńczone egzaminem i pracą dyplomową. Normatywny i maksymalny czas studiów wynoszący 8 oraz 10 semestrów jest w praktyce systematycznie przekraczany.

W Politechnice Monachijskiej student informatyki uzyskuje dyplom przeciętnie po 13 semestrach. W okresie tym poznaje cztery podstawowe obszary wiedzy informatycznej: informatyki teoretycznej, technicznej, praktycznej oraz aplikacyjnej.

Informatyka teoretyczna wchodzi w sposób matematyczny w zagadnienia rozwiązywalności problemów (algorytmy, algebra liniowa, analiza, logika, teoria automatów, teoria języków formalnych, teoria złożoności).

Informatyka techniczna obejmuje fizyczne i elektroniczne podstawy konstrukcji i sposobu działania komputerów oraz ich urządzeń peryferyjnych, a więc całą problematykę sprzętu.

Informatyka praktyczna zajmuje najwięcej miejsca, ponieważ odgrywać będzie decydującą rolę w pracy zawodowej. Obejmuje bardzo rozległą wiedzę o systemach operacyjnych, językach programowania oraz standardowych algorytmach (różne strategie rozwiązań). Wykłady są uzupełnione ćwiczeniami z zakresu projektowania i uruchamiania programów budowy systemów operacyjnych oraz opanowania umiejętności posługiwania się różnymi językami programowania.

Informatyka aplikacyjna umożliwia wykorzystanie wiedzy pozyskanej w trzech wyżej omówionych obszarach studiów do rozwiązywania problemu w wybranej dziedzinie zastosowań. Typowymi takimi dziedzinami najczęściej są podstawowe dyscypliny naukowe: matematyka, fizyka, elektrotechnika, nauki ekonomiczne, prawo, medycyna.

Informatyk dyplomowany, po przebrnięciu przez tak wszechstronny program studiów, może sprostać w przyszłej działalności zawodowej bardzo szerokiej skali wymagań. Może np. prowadzić badania teoretyczne i stosowane w różnych obszarach informatyki i jej zastosowań, a także odgrywać wiodącą rolę w projektowaniu, wyborze, adaptacji i wdrażaniu metod, technologii i systemów komputerowych zarówno znanych, jak i jeszcze nierozpoznanych obszarach zastosowań informatyki. Jest on również bardzo poszukiwanym specjalistą przy czynnościach sprzedaży systemów komputerowych lub doradzaniu użytkownikom w wyborze, projektowaniu, instalowaniu i eksploatacji sprzętu komputerowego i oprogramowania, a także przy konserwacji i rozwoju systemów już eksploatowanych. W pracy zawodowej szybko osiąga stanowiska kierownika ośrodka obliczeniowego, działu przetwarzania lub zastosowań, a także kierownika realizacji dużych projektów informatycznych.

Informatyk dyplomowany jest również bardzo poszukiwanym kandydatem w dziedzinie projektowania i realizacji kursów szkolenia i doskonalenia zawodowego u producentów sprzętu i oprogramowania, w instytucjach oświaty zawodowej oraz w firmach doradztwa informatycznego.

Przeprowadzona przez zachodnioniemieckie Towarzystwo Informatyczne (GI – *Gesellschaft für Informatik*) ankieta wykazała, że ok. 30% informatyków dyplomowanych pracuje w szkołach wyższych i instytucjach naukowych, 36% w szeroko pojętym przemyśle informatycznym, 27% u użytkowników sprzętu komputerowego, a 5,5% wykonuje informatykę jako wolny zawód. Jeśli chodzi o zarobki dyplomowanych informatyków to są one bardzo niechętnie ujawniane, zgodnie z powszech-

nie obowiązującym na Zachodzie podejściem do tego rodzaju informacji traktowanych jako sferę całkowicie osobistą. Z ogólnikowych wypowiedzi, jakie udało się uzyskać od kierownika firmy wytwarzającej oprogramowanie, wynika, że początkowo płaca absolwenta z dobrą znajomością systemu Unix oraz języka C wynosi rocznie ok. 68 000 marek (ok. 40 000 dolarów USA), oczywiście brutto, a więc przed opłaceniem podatku i ubezpieczenia. Po 6–7 latach pracy pracownik taki osiąga zwykle stanowisko kierownika zespołu i wynagrodzenie jego ulega podwojeniu.

Jeśli chodzi o udział kobiet na studiach informatycznych, to we wspomnianej już Politechnice Monachijskiej wynosi on obecnie ok. 13%. Dominacja mężczyzn jest głównie wynikiem masowego zainteresowania, a nawet fascynacji chłopców komputerami w okresie szkolnym. Absolwentki studiów informatycznych są jednak bardzo cenione i poszukiwane, zwłaszcza do pracy zespołowej, w której są bardziej zdyscyplinowane od mężczyzn.

Wielu absolwentów wyraża jednak wątpliwości, czy w większości przypadków do kariery informatycznej są konieczne studia wyższe, ponieważ w pracy zawodowej stwierdzają fakty posługiwania się stosunkowo wąskim zakresem wiedzy, którą przecież można zdobyć w znacznie krótszym czasie i niewspółmiernie mniejszym wysiłkiem.

(WK)

Nowe systemy RISC

Wszystko wskazuje na to, że w ostatnim okresie w konkurencji pomiędzy układami o złożonej liście rozkazów (CISC – *Complex Instruction Set Computation*) i zredukowanej (RISC – *Reduced Instruction Set Computation*) zwyciężają te ostatnie. Mikroprocesory, takie jak Intel 80680, MIPS R3000, Clipper C100 i C300, AMD Am 29000 i Motorola 88000 pozwalają na realizację systemów o coraz większych możliwościach. Chcąc wymienić podstawowe cechy układów RISC należy wspomnieć o rozdziale obliczeń. Poprzednio, by zaoszczędzić drogiej pamięci, chciano w każdym rozkazie zrealizować jak najwięcej obliczeń. Wraz z obniżeniem cen pamięci stało się to mniej istotne. Jeśli w ciągu każdego rozkazu wykonywana jest jedna operacja, kompilator może reorganizować rozkazy, eliminować powtarzające się obliczenia adresów, przechowywać wyniki pośrednie, zamieniać mnożenia na dodawania itp. Umożliwia to znaczne zwiększenie wydajności. Ponieważ w systemie RISC każdy rozkaz wymaga takiej samej kolejności czynności, operacje te można zrównoleglić sprzętowo (ang. *pipelining*) i cały rozkaz zostanie wykonany w jednym cyklu zegarowym.

Firma MIPS Computer Systems najpierw opracowała kompilatory, analizując w tym celu szereg programów aplikacyjnych. Następnie zrealizowano sprzęt z myślą o jak najszybszym wykonywaniu funkcji podstawowych, co pozwoliło inne funkcje realizować programowo. Identyczne postępowanie zastosowano przy opracowywaniu zespołu rejestrów. Procesor centralny zawiera 32 rejestry 32-bitowe, a procesor zmiennego przecinka – 16 rejestrów 64-bitowych wraz z 64 połączeniami danych do szybkich obliczeń o podwójnej dokładności. Procesor R3000 zapewnia połączenia do operacji równoległych, zespół układów funkcyjnych, jednostkę arytmetyczno-logiczną, 32-bitowy układ przesuwający (ang. *shifter*) i autonomiczny układ mnożąco-dzielący, a także zarządzanie pamięcią i układ operacji specjalnych. 64-wejściowy bufor translacji adresu, w pełni asocjacyjny, zamienia w ciągu 20 ns adres wirtualny na adres fizyczny. Systemy w tej architekturze wymagają mało układów pomocniczych, gdyż większość układów interfejsu zawarta jest w kostce. W układzie podwójnej pamięci pomocniczej (ang. *dual-cache*) uzyskuje się szybkość przesyłania 200 MB/s przy zegarze 25 MHz. Firma oferuje też bogaty zestaw narzędzi sprzętowo-programowych do projektowania systemów.

System Clipper firmy Intergraph był pionierskim układem RISC choć zawierał już zintegrowane układy zmiennego przecinka i pamięci pomocniczej. Według danych opublikowanych w listopadzie 1989 roku, istniało w tym czasie ponad 250 zastosowań programowych i ponad 27 tysięcy systemów użytkowych wykorzystywanych jako stacje robocze, serwery, wieloprocesorowe bazy danych, układy edytorskie, pisaki i układy przetwarzania obrazów w czasie rzeczywistym. Posługują się one różnymi magistralami, takimi jak PC/ATbus, VMEbus i Nubus. Opracowywane są wersje unowocześnione zarówno w technice ECL jak i CMOS, gdzie liczba cykli zegarowych na rozkaz zmniejszy się z 4 do 1,2–1,4.

Firma Sun Microsystems, znany projektant i dostawca stacji roboczych, opracowała architekturę SPARC (od *Scalable Processor Architecture*), którą zaoferowała innym wytwórcom jako standard otwarty. Zastosowano tu różne technologie BiCMOS, ECL i GaAs osiągając wydajności od 10 do setek operacji na sekundę. W miarę rozwoju systemu, wzrosła liczba realizowanych funkcji, a malała liczba stosowanych kostek. Specyfiką tego rozwiązania są tzw. okna nakładających się rejestrów, działające jako lokalne, zmienne pamięci pomocnicze. Architektura ta może działać pod systemem operacyjnym SunOS, opartym między innymi na Unix System V. Inne oprogramowanie pozwala na posługiwanie się różnymi językami wysokiego poziomu. **(JR)**

Nieprawidłowe wykorzystanie sprzętu w RFN

W czasopismach informatycznych RFN pojawiają się coraz częściej artykuły na temat złego wykorzystania sprzętu komputerowego w przedsiębiorstwach. Dotyczy to najliczniejszej grupy użytkowników, jaką obecnie stanowią małe przedsiębiorstwa. Za podstawową przyczynę tego zjawiska uważa się złe przygotowanie zawodowe personelu obsługi komputerów, co przypisuje się w pierwszym rzędzie typowym skłonnościom właścicieli przedsiębiorstw tej kategorii do maksymalnego ograniczania kosztów, w tym przypadku ponoszonych na szkolenie personelu.

Badając głębiej to alarmujące zjawisko, wielu komentatorów stwierdza jednak, że w równym, jeśli nie większym stopniu, winę ponoszą dostawcy komputerów, którzy zamiast efektywnych instrumentów szkolenia, jakimi są obecnie komputerowe programy dydaktyczne, przekazują swym klientom jedynie standardowe podręczniki operatorskiej obsługi sprzętu, napisane zazwyczaj w sposób mało zrozumiały dla osób bez profesjonalnego przygotowania informatycznego. Tymczasem w małych przedsiębiorstwach, stosujących wyłącznie komputery osobiste i to do bardzo prostych, standardowych zastosowań w sferze zarządzania, zatrudnianie informatyków jest nieuzasadnione.

Niestety dostarczenie nawet najlepszego programu dydaktycznego problemu nie rozwiązuje, ponieważ przyswojenie jego treści oraz opanowanie niezbędnego zakresu praktycznych umiejętności wymaga zarówno czasu, jak i znacznego wysiłku intelektualnego ze strony pracownika. Oba te warunki są w małych przedsiębiorstwach bardzo trudne do spełnienia. Wytypowany do przeszkolenia pracownik jest nadal obciążony obowiązkami wynikającymi z dotychczasowej technologii pracy, ponieważ nikt nie może go w tym zastąpić, a przedsiębiorstwo nie może ani na moment przerwać swojej działalności. W tej sytuacji pracownik może przeznaczyć na przeszkolenie wyłącznie godziny pozasłużbowe, co właściciel teoretycznie powinien mu zrekompensować. W praktyce często tak nie jest, ponieważ właściciel każde szkolenie traktuje jako podnoszenie przez pracownika kwalifikacji zawodowych, nieodzowne w warunkach gospodarki rynkowej dla utrzymania się na dotychczasowym stanowisku.

Warto zauważyć, że tego rodzaju wyrachowane podejście, uzasadnione twardymi regułami walki o utrzymanie istnienia małego przedsiębiorstwa, nie występuje w dużych przedsiębiorstwach. W jednostkach takich z reguły planowane są środki finansowe i warunki organizacyjne dla regularnego szkolenia i doszkalania personelu. Traktowane jest to jako niezbędne przestawienie technologiczne (podobnie jak np. w produkcji) i odbywa się, podobnie jak u nas, z pełnym, a co najmniej częściowym oderwaniem od pracy. **(WK)**

Królikowski Z.: Jednoczesne przetwarzanie wielu zapytań w relacyjnych bazach danych

INFORMATYKA 1990, nr 6, s. 1

Algorytm jednoczesnego przetwarzania wielu zapytań w systemach zarządzania relacyjną bazą danych, zapewniający w tym obszarze funkcji znaczny wzrost efektywności działania systemu.

Królikowski Z.: Simultanous processing of many queries in relational data bases

INFORMATYKA 1990, No. 6, p. 1

The simultanous processing algorithm for many queries in relational data base management systems, which assures in this functions area substantial operation effectivity increase of the system.

Królikowski Z.: Gleichzeitige Verarbeitung von mehreren Abfragen in Relationsdatenbanken

INFORMATYKA 1990, Nr. 6, S. 1

Algorithmus für gleichzeitige Verarbeitung von mehreren Abfragen in Verwaltungssystemen der Relationsdatenbank, der in diesem Funktionsbereich erhebliche Effektivitätssteigerung des Systembetriebes sichert.

Orwaldi P.: Metody programowania pamięci EPROM

INFORMATYKA 1990 nr 6, s. 5

Charakterystyka konstrukcji i technologii wytwarzania pamięci stałych oraz omówienie różnych sposobów programowania pamięci reprogramowalnych EPROM.

Orwaldi P.: Programming methods for EPROM storage

INFORMATYKA 1990, No. 6, p. 5

Characteristics of the permanent storage construction and manufacturing technology and discussion of different programming methods for reprogramable EPROM storage.

Orwaldi P.: Programmierungsmethoden eines EPROM-Speichers

INFORMATYKA 1990, Nr. 6, S. 5

Eine Charakteristik von Konstruktion und Technologie der Festspeicherherstellung und eine Besprechung von verschiedenen Programmierungsmethoden der reprogrammierbaren EPROM-Speicher.

Dawidowski J.: Ręczne rejestratory danych

INFORMATYKA 1990, nr 6, s. 8

Geneza rozwoju, charakterystyka techniczna i użytkowa oraz porównanie najczęściej stosowanych modeli ręcznych rejestratorów danych.

Dawidowski J.: Hand-held data recorders

INFORMATYKA 1990, No. 6, p. 8

Development history, technical and operational characteristics, as well as comparison of the most often applied models of hand-held data recorders.

Dawidowski J.: Tragbare Datenerfassungsgeräte

INFORMATYKA 1990, Nr. 6, S. 8

Entwiklungsgeschichte, technische und benutzerorientierte Charakteristik, sowie ein Vergleich der am meisten angewendeten Modelle von tragbaren Datenerfassungsgeräten.

Kłopotek M.: Rozpoznawanie słów w mowie ciągłej

INFORMATYKA 1990, nr 6, s. 12

Charakterystyka algorytmów oraz stosowanych metod realizacji automatycznego rozpoznawania słów w mowie ciągłej.

Kłopotek M.: Word recognition in continous speech

INFORMATYKA 1990, No. 6, p. 12

Characteristics of algorithms and applied methods for automatized word recognition in continous speech.

Kłopotek M.: Worterkennung in kontuierlicher Sprache

INFORMATYKA 1990, Nr. 6, S. 12

Eine Charakteristik von Algorithmen und angewendeten Methoden für Realisierung der automatisierten Worterkennung in kontinuierlicher Sprache.

Żakowicz T.: CDS/ISIS – narzędzie do zarządzania bazami danych (2)

INFORMATYKA 1990, nr 6, s. 15

Druga część charakterystyki najnowszej wersji pakietu programowego CDS/ISIS, zawierająca omówienie sposobu użytkowania bazy danych w trybie bezpośrednim, zarządzania podstawowymi funkcjami systemu baz danych oraz możliwości rozszerzenia tego pakietu.

Żakowicz T.: CDS/ISIS – a tool for data base management (2)

INFORMATYKA 1990, No. 6, p. 15

Second part of characteristics of the most recent version of the CDS/ISIS program package, which includes discussion on the data base using method in the direct mode, management of basic functions of the data base system, as well as on extension possibilities of the package.

Żakowicz T.: CDS/ISIS – ein Werkzeug zur Datenbankverwaltung (2)

INFORMATYKA 1990, Nr. 6, S. 15

Zweiter Teil einer Charakteristik von neuester Version des CDS/ISIS-Programmpakets, der eine Besprechung von Gebrauchsmethode der Datenbank in Direktmodusbetrieb, Verwaltung der Grundfunktionen von Datenbanksystem, sowie Erweiterungsmöglichkeiten dieses Pakets umfasst.

Malec J.: Maszyny lispowe (2)

INFORMATYKA 1990, nr 6, s. 19

Druga część charakterystyki maszyn lispowych, zawierająca omówienie najbardziej popularnych modeli tych maszyn.

Malec J.: Lisp machines (2)

INFORMATYKA 1990, No. 6, p. 19

Second part of characteristics of the Lisp machines, which includes discussion on most popular models of these machines.

Malec J.: Lisp-Maschinen (2)

INFORMATYKA 1990, Nr 6, S. 19

Zweiter Teil einer Charakteristik von Lisp-Maschinen, der eine Besprechung von populärsten Modellen dieser Maschinen umfasst.

Dembiński P.: Protokoły komunikacyjne (2)

INFORMATYKA 1990, nr 6, s. 24

Druga część poglądowego omówienia istoty oraz funkcji protokołów w sieciach komputerowych o architekturze warstwowej, zawierająca omówienie modelu warstwowego ISO oraz sformalizowanych metod specyfikacji protokołów.

Dembiński P.: Communication protocols (2)

INFORMATYKA 1990, No. 6, p. 24

Second part of the demonstrative discussion on protocol essence and functions in computer networks with layer architecture, which includes discussion on ISO model and formalized protocol specification methods.

Dembiński P.: Kommunikationsprotokolle (2)

INFORMATYKA 1990, Nr. 6, S. 24

Zweiter Teil einer anschaulichen Besprechung des Wesens und Funktionen von Protokellen in Computernetzen mit -Schichtsmodells und der formalisierten Methoden von Schichtsmodells und der formalisierten Methoden von Protokollspezifikation umfasst.

Bezpieczeństwo danych

Problematyka bezpieczeństwa danych, a zwłaszcza metod ich zabezpieczania przed dostępem przez osoby nieupoważnione pojawia się coraz częściej w zagranicznych czasopismach informatycznych. Wraz z rozwojem komputeryzacji, a zwłaszcza z jej upowszechnieniem, pojawia się bowiem specyficzna przestępczość, polegająca na kradzieży lub rozmyślnym niszczeniu informacji istotnych dla działalności przedsiębiorstwa. Rewolucja mikrokomputerowa oraz masowość przesyłania danych za pośrednictwem publicznych sieci telefonicznych stworzyły szczególnie sprzyjające warunki do rozprzestrzenienia się tego rodzaju przestępczości (masowy dostęp do sprzętu komputerowego, uproszczenie i ułatwienie dostępu do gigantycznych zasobów informacji). Dlatego bezpieczeństwo danych stało się jednym z kluczowych problemów technologii informatycznej.

Podstawowym źródłem zagrożeń stała się zewnętrzna pamięć komputera, której pojemność wzrosła do granic uznanych jeszcze wczoraj za fantazję. Pamięć taką można spotkać nie tylko w dobrze strzeżonych ośrodkach obliczeniowych, ale również w wyposażeniu powszechnie dostępnych komputerów osobistych. Skalę zagrożenia doskonale ilustruje stosowany już w tej klasie sprzętu dysk sztywny o pojemności 600 MB, będącej równoważnikiem ok. 300 tys. znormalizowanych stron maszynopisu. Przechowanie takiej liczby dokumentów papierowych wymaga w archiwum ok. 21 mb półek (4–5 typowych szaf biurowych). Tak dużą masę dokumentów można wywieźć jedynie samochodem ciężarowym. Skopiowanie całej zawartości dysku wymaga natomiast zaledwie 500 standardowych dyskietek (po 1,2 MB), które można bardzo łatwo wynieść, ponieważ mieszczą się bez trudu w jednej, a najwyżej w dwu teczkach do przewożenia akt.

Zagrożenia bezpieczeństwa danych może nastąpić nie tylko przez ich wykradzenie, ale również przez zniszczenie lub zniekształcenie za pomocą umyślnie wprowadzonych (najczęściej za pośrednictwem publicznej sieci telefonicznej) specjalnych programów niszczących (tzw. *killer-programs*, programy-zabójcy).

Przedmiotem działań przestępczych, ukierunkowanych na kradzież lub niszczenie danych są w warunkach gospodarki rynkowej i związanej z nią walki konkurencyjnej wszelkie tajemnice technologiczne, organizacyjne lub handlowe przedsiębiorstw.

Potencjalni złodzieje danych istnieją praktycznie w każdym przedsiębiorstwie. Jego właściciela, który sądzi, że doskonale zna swoich współpracowników i dlatego może polegać na ich lojalności, często spotyka przykre rozczarowanie. Należy bowiem zawsze liczyć się z tym, że w firmie znajdą się niezadowoleni pracownicy, pragnący zemścić się na pracodawcy, a nawet jego koledzy, którzy planując zmianę miejsca pracy postanawiają przechwycić najcenniejsze informacje celem stworzenia sobie lepszej pozycji u nowego pracodawcy.

Właściwe działania organizacyjne mogą zmniejszyć niebezpieczeństwo, a nawet zapobiec wspomnianym zagrożeniom. Wymaga to ścisłego określenia zarówno uprawnień dostępu do pomieszczeń, w których są zainstalowane komputery, jak i zabezpieczenia ich przed ewentualnym wykorzystaniem przez osoby nieupoważnione. Pierwszy warunek mogą oczywiście spełnić tylko duże przedsiębiorstwa. Stosowane są w tym celu różne rozwiązania: od odczytywanych automatycznie dowodów osobistych do pełnego zabezpieczenia obiektu przez wyspecjalizowane firmy.

Podobnie wygląda sprawa ochrony sprzętu komputerowego przed dostępem osób nieupoważnionych. Minimum ochrony to dostęp do komputera tylko po wprowadzeniu przez użytkownika zastrzeżonego hasła lub kodu. Dalszym stopniem zabezpieczenia jest ograniczenie dostępu jedynie do określonych zbiorów danych.

Wszystkie tego rodzaju środki i metody muszą być wprowadzane w uzgodnieniu z zainteresowanymi pracownikami. Oznacza to konieczność jednoznacznego określenia danych uznanych za tajemnicę firmy. Współdziałanie w tym zakresie zarówno pracowników niższych szczebli zarządzania, jak i kadry kierowniczej, prowadzi do pogłębienia więzi pracowników z firmą, które Amerykanie określają jako identyfikowanie się z jej interesami (ang. *corporate identity*).

Chociaż w Polsce nie wystąpiły jeszcze spory sądowe o kradzież danych, to jednak w konsekwencji zaostrzającej się walki konkurencyjnej na rynku, należy spodziewać się ich rychłego pojawienia.

Walka z kradzieżą danych wymaga oczywiście ustawowego zdefiniowania pojęcia tajemnicy przedsiębiorstwa z punktu widzenia reguł gospodarki rynkowej. Można przypuszczać z dużym prawdopodobieństwem, że definicja taka istniała w naszym przedwojennym, wówczas bardzo nowoczesnym, ustawodawstwie gospodarczym. Zanim jednak ustawodawstwo to zacznie ponownie funkcjonować, warto aby nasi pracownicy zapoznali się w tym względzie z praktyką RFN, gdzie sprawy kradzieży danych coraz częściej trafiają do sądów. Definicję tajemnicy przedsiębiorstwa rozstrzygnął tam ostatecznie Federalny Trybunał Administracyjny. Stwierdził on, że tajemnicą taką jest każdy fakt, który spełnia następujące trzy warunki: 1) dotyczy wyłącznie własnego przedsiębiorstwa, 2) znany jest tylko ograniczonemu kręgowi osób (nie jest powszechnie dostępny), 3) wolą właściciela przedsiębiorstwa ma być utrzymany w tajemnicy, jednak pod warunkiem, że zachowanie takie jest uzasadnione interesem gospodarczym.

dokończenie na s. 18

PRZEDSIĘBIORSTWO
ZASTOSOWAŃ INFORMATYKI

meditronik

oferuje:

- systemy mikrokomputerowe
- programy aplikacyjne dla różnych dziedzin gospodarki (na życzenie wysyłamy katalog)
- poszukiwane komponenty elektroniczne
- interfejs do kamery video (opc. CCD) z bogatą biblioteką oprogramowania
- emulator Z80
- tester układów scalonych i pamięci
- programator BPROM
- asynchroniczny procesor komunikacyjny
- konwerter RS-232 – Centronics

instaluje:

- połączenia międzykomputerowe (XT/AT – ODRA/RIAD/IBM)
- systemy sieciowe (NOVELL)
- systemy wielodostępne (SCO Xenix 286, 386, Unix System V)

Jeżeli jesteś autorem oryginalnego programu aplikacyjnego

- skontaktuj się z nami, będziemy pośredniczyć w sprzedaży Twojego programu, dbając o ochronę Twoich praw autorskich!

Nasz adres:

00-194 Warszawa, ul. Dzika 4
tel. (02) 635-22-63, 635-22-64,
fax (02) 635-21-95
tlx 816075 medi pl

EO/605/89

Ogłoszenie w INFORMATYCE bezbłędną szansą trafienia do właściwego odbiorcy!

Miesięcznik INFORMATYKA jest od 25 lat **jedynym** w Polsce czasopismem naukowo-technicznym poświęconym wszystkim aspektom technologii informatycznej (sprzęt komputerowy, oprogramowanie, zastosowania). Taka pozycja czasopisma sprawia, że INFORMATYKA dociera nie tylko do najszerszych kręgów środowiska profesjonalnego, ale również do wszystkich zainteresowanych aktualnymi możliwościami tej technologii.

Ogłoszenie w INFORMATYCE to najskuteczniejszy sposób pozyskania nowych odbiorców i rozszerzenia rynku zbytu. Gwarantujemy Państwu największe efekty przy najniższych nakładach finansowych, ponieważ **ceny ogłoszeń w INFORMATYCE są znacznie niższe** niż w innych czasopismach:

• cała strona	500 000 zł
• 3/4 strony	450 000 zł
• 2/3 strony	400 000 zł
• 1/2 strony	300 000 zł
• 1/3 strony	200 000 zł
• 1/4 strony	150 000 zł
• 1/8 strony	100 000 zł
• poniżej 1/8 strony	2 000 zł/cm^2

Dopłaty:

• za dodatkowy kolor	+ 30%
• za I i IV stronę okładki	+ 100%
• za II i III stronę okładki	+ 50%

Rabaty:

• za powtórzenie 3–6 razy	– 10%
• za powtórzenie ponad 6 razy	– 20%
• za teksty o charakterze informacji techniczno-handlowej (artykuły)	– 50%

Zapewniamy szybką i terminową realizację ogłoszeń!

Nasz adres: INFORMATYKA, 01–552 Warszawa, Pl. Inwalidów 10. p. 128, tel. 39–14–34